Late Fusion Multiple Kernel Clustering Refined via Optimal Linear Graph Filtering

Henghui Jiang[1], Yiqing Guo[1], Yan Chen[2], and Liang Du[1(✉)]

[1] Shanxi University, Taiyuan, 030006, China
{jianghenghui, guoyiqing, duliang}@sxu.edu.cn
[2] Taiyuan University of Technology, Taiyuan, 030600, China
{yanchen01}@tyut.edu.cn

Abstract. Multiple Kernel Clustering (MKC) aims to improve clustering performance by integrating complementary information from candidate kernels. Among existing MKC methods, late fusion MKC (LFMKC) offers superior scalability by aggregating clustering partitions rather than full kernel matrices. However, its reliance on fixed base partitions often leads to suboptimal representations and degraded clustering performance. To address this, we propose Late Fusion MKC Refined via Optimal Linear Graph Filtering (LFMKC-OLF), which enhances partition quality while preserving linear computational complexity. Specifically, bipartite graphs are constructed for each base partition, upon which high-order low-pass filters based on heat kernel diffusion and complementary high-pass filters are designed to capture both global consistency and fine-grained structural details. A consensus filtering mechanism is introduced by optimally combining view-specific filters to refine multi-scale representations. Furthermore, a joint clustering objective integrates both smoothed and detail-enhanced partitions to effectively mitigate over-smoothing. Extensive experiments on eight benchmark datasets demonstrate that LFMKC-OLF consistently outperforms 12 state-of-the-art LFMKC methods, while maintaining high computational efficiency for large-scale clustering tasks. Our code is publicly available at: `https://github.com/sxuHui/LFMKC-OLF`.

Keywords: Multiple Kernel Clustering · Graph filter · Over-smoothing

1 Introduction

Multiple Kernel Clustering (MKC) is a powerful machine learning technique for handling complex, non-linearly separable data by mapping it into higher-dimensional Hilbert spaces[19,20,32]. In the era of big data, where information often collects from multiple views or sources, MKC integrates multi-source data within the kernel space to assign samples to distinct clusters. By learning an optimal combination from a predefined set of kernels, MKC effectively captures complementary information across diverse feature representations, thereby improving clustering robustness and adaptability in multi-view scenarios.

According to the stage of information fusion, existing MKC methods can be broadly divided into two categories: early kernel fusion and late partition fusion methods [27]. The first convert kernel information into a set of predefined kernel matrices, each of size $n \times n$, with the primary goal of optimally combining these matrices to learn the consensus clustering information. The other category focus on enhancing clustering performance and mitigating the impact of noise by applying fusion at the partition level [21]. By fusing the information of individual partitions to obtain the underlying shared representation, late fusion-based MKC (LFMKC) significantly reducing the computational burden, demonstrating promising performance across various applications [21].

Despite recent progress, LFMKC still faces critical limitations. First, its reliance on fixed base partitions imposes a performance bottleneck, as these initial representations often fail to capture the underlying data structure adequately. Moreover, prevailing methods predominantly focus on optimizing kernel weights to exploit inter-view consistency and complementarity, yet they largely neglect the high-order correlations across views, thus restricting the integration of richer multi-kernel information. Additionally, while low-pass graph filters have been employed to enhance intra-cluster coherence by suppressing noise, excessive filtering inevitably leads to over-smoothing, where crucial high-frequency details are erased and cluster boundaries become indistinct, ultimately undermining clustering performance.

To tackle these challenges while maintaining linear computational complexity, we propose a novel method termed Late Fusion Multiple Kernel Clustering Refined via Optimal Linear Graph Filtering (LFMKC-OLF). The approach begins by constructing bipartite graphs for each kernel partition, from which we design view-independent low-pass and high-pass graph filters to capture complementary low- and high-frequency signals respectively. Specifically, these low-pass filters derived from diffusion processes capture high-order relationships and smooth partitions, revealing the underlying cluster structures; while high-pass filters preserve critical high-frequency signals encoding fine-grained cluster boundaries. To integrate multi-view information effectively, we introduce a consensus filter learning strategy, where the final low-pass and high-pass filters are obtained as optimal linear combinations of the individual per-view filters. To efficiently handle large-scale datasets, we exploit the low-rank structure inherent in bipartite graph-induced filters, enabling a linear-time algorithm for filtering. Crucially, we propose a unified optimization framework that considers the joint clustering loss under smoothed partitions and detail-enhanced partitions. By jointly modeling low-pass and high-pass responses, our framework captures both smooth cluster structures and fine-grained boundary details, effectively mitigating the cratical information loss caused by over-smoothing. To solve the resulting problem, we develop an iterative algorithm that updates all variables with linear complexity. Our contributions are summarized as follows:

- We propose a novel LFMKC method that integrates bipartite graph-induced high-order filters based on heat kernel diffusion. By optimizing the lin-

ear combination of these filters, our approach promotes clustering friendly smooth representation and effectively refines the base partitions.
– We propose a novel framework that jointly clustering both on smoothed and detail-enhanced partitions by leveraging the multi-scale representation of graph filtering. By integrating critical detail structural information into the smoothed embedding, it effectively mitigates over-smoothing and achieves optimal filtering refinement.
– Extensive evaluations on eight diverse datasets reveal that LFMKC-OLF not only surpasses 12 cutting-edge LFMKC methods in clustering performance but also features a linear-time iterative optimization algorithm that enables efficient training and inference on large-scale data, showcasing its practicality for applications.

2 Related Work

2.1 Late Fusion Multiple Kernel Clustering

LFMKC has garnered significant attention for its efficiency and promising performance. The LFMKC framework typically proceeds in two stages: generating base partitions from each kernel—typically derived from the top eigenvectors—and subsequently fusing these partitions into a consensus result. Recent efforts have explored diverse strategies to enhance the fusion stage. For example, Wang et al. [21] propose a strategy that maximizes the alignment between weighted base partitions to learn a consensus partition. Liu et al. [16] propose to decompose the consensus partition into cluster labels and centroids, without requiring further post-processing. Li et al. [11] integrate the min-max optimization framework from [15] into LFMKC to refine the fusion process. Yang et al. [27] reconstruct kernel partitions using a self-expression strategy guided by refined similarity graphs to capture more complex relationships among partitions. Wu et al. [23] employ the Grassmann manifold for partitions fusion, maintaining topological structure in high-dimensional space. Zhang et al. [31] incorporate sample weights and introduce a prior-informed regularization term to enhance fusion stability. However, these methods heavily rely on the quality of the original base partitions and fail to fully exploit high-order information, which limits their effectiveness in complex or noisy scenarios.

2.2 Graph Filters

Graph filters play a crucial role in graph signal processing by enabling the manipulation of signals defined on graph structures, leveraging adjacency relationships between nodes. These filters are employed for tasks such as signal smoothing, noise reduction, and feature extraction [9]. Recent studies have explored integrating low-pass graph filters to enhance data representations in multi-view clustering (MVC). Lin and Kang [14], Chen et al. [2] and Kang et al. [10] adopt a two-stage strategy: constructing low-pass filters to smooth the original features

before applying MVC. Alternatively, Zhou et al. [33], and Zhou and Du [34] propose learning similarity graphs in the node domain, adjusting graph filters dynamically during clustering to improve adaptability. Yang et al. [26] examine the convex combination of multiple graph filters, combining their strengths to improve filtering effectiveness. Guo et al. [7] further build on this by using bipartite graphs to construct filters and achieve linear-time filtering via the Woodbury formula. Despite progress in the utilization of graph filtering, existing methods predominantly focus on low-frequency components to capture smooth representations, often indiscriminately discarding high-frequency signals as noise. This can inevitably result in over-smoothing and impair the ability to preserve discriminative node features. Although some methods [14,10,33,34] attempt mitigation by heuristically constraining the filter order within an empirical range, such strategies still restrict the exploitation of high-order information and hinder the pursuit of optimal filtering.

3 Proposed Method

In this section, we present the proposed method, starting with the base partitions $\{\mathbf{H}_v\}_{v=1}^{V} \in \mathbb{R}^{n\times d}$, where V denotes the number of views, n the number of samples, and d the embedding dimension. Each $\mathbf{H}_v$ consists of orthogonal vectors obtained via eigenvalue decomposition of the corresponding kernel matrix.

3.1 View-independent Bipartite Graph Construction

To reduce computational overhead, we introduce bipartite graphs that capture affinities between samples and a reduced set of representative anchors. Prior work has shown that diversifying anchor graphs can significantly enhance clustering performance [28]. Motivated by this, we adopt a view-independent anchor selection strategy. Specifically, we first perform Kmeans clustering independently on each partition $\mathbf{H}_v$ to obtain m_v centroids, forming the anchor matrix $\mathbf{O}_v \in \mathbb{R}^{m_v\times d}$. The bipartite graph $\mathbf{Z}_v \in \mathbb{R}^{n\times m_v}$ is then learned by solving the following optimization problem:

$$\min_{\mathbf{z}_i^v\mathbf{1}=1,\mathbf{z}_i^v\geq 0} \sum_{j=1}^{m_v} \mathrm{dist}(\mathbf{h}_i^v, \mathbf{o}_j^v) z_{ij}^v + \tau \sum_{j=1}^{m_v} (z_{ij}^v)^2, \tag{1}$$

where $\mathbf{z}_i^v$ is the i-th row of $\mathbf{Z}_v$, $\mathrm{dist}(\mathbf{h}_i^v, \mathbf{o}_j^v) = \|\mathbf{h}_i^v - \mathbf{o}_j^v\|_2^2$ denotes the squared euclidean distance between the i-th sample and j-th anchor, and τ is a regularization hyperparameter controlling the smoothness of the assignment. To further enhance scalability, we impose sparsity on $\mathbf{Z}_v$ by restricting each sample to connect only with its k nearest anchors, following the strategy in [18], with k fixed to 5 throughout our experiments. For each sample i, assume that the distances $\mathrm{dist}(\mathbf{h}_i^v, \mathbf{o}_1^v), \mathrm{dist}(\mathbf{h}_i^v, \mathbf{o}_2^v), \ldots, \mathrm{dist}(\mathbf{h}_i^v, \mathbf{o}_{m_v}^v)$ are sorted in ascending order.

Problem (1) admits a closed-form solution given by:

$$z_{ij}^v = \begin{cases} \frac{\mathrm{dist}(\mathbf{h}_i^v, \mathbf{o}_{k+1}^v) - \mathrm{dist}(\mathbf{h}_i^v, \mathbf{o}_j^v)}{k\mathrm{dist}(\mathbf{h}_i^v, \mathbf{o}_{k+1}^v) - \sum_{k'=1}^{k} \mathrm{dist}(\mathbf{h}_i^v, \mathbf{o}_{k'}^v)}, & \text{if } j \leq k, \\ 0, & \text{if } j > k. \end{cases}$$

Subsequently, the similarity matrix $\mathbf{S}_v \in \mathbb{R}^{n \times n}$ is constructed from the bipartite graph $\mathbf{Z}_v$ to capture indirect associations between samples. Specifically, we normalize $\mathbf{Z}_v$ by computing:

$$\mathbf{P}_v = \mathbf{Z}_v \mathbf{\Delta}_v^{-\frac{1}{2}}, \tag{2}$$

where $\mathbf{\Delta}_v$ is a diagonal matrix with entries $\Delta_{jj}^v = \sum_{i=1}^{n} \mathbf{Z}_{ij}^v$. The final similarity matrix is then obtained as:

$$\mathbf{S}_v = \mathbf{P}_v \mathbf{P}_v^\top, \tag{3}$$

By construction, $\mathbf{S}_v$ is a doubly stochastic matrix, i.e., each row and column sums to one, enabling favorable properties for subsequent graph filtering and clustering steps.

3.2 Smoothed Partition Fusion with Low-pass Filtering

Although similarity graphs $\{\mathbf{S}_v\}_{v=1}^V$ effectively capture local structure, they overlook higher-order neighborhood relationships, missing global dependencies between distant samples. To address this, we adopt heat kernel diffusion [13] to integrate similarity graphs across all orders. Formally, the diffusion process is expressed as:

$$\mathbf{G}_v = e^{-\eta} \sum_{t=0}^{\infty} \frac{\eta^t}{t!} (\mathbf{S}_v)^t = \exp(-\eta \mathbf{L}_v), \tag{4}$$

where $\eta \geq 0$ controls the decay rate [3] and $\mathbf{L}_v = \mathbf{I} - \mathbf{S}_v$ denotes the normalized Laplacian. The resulting $\mathbf{G}_v$ acts as a low-pass filter that suppresses high-frequency components, effectively reducing noise and enhancing global structural patterns.

To fully explore the information across views and enhance global consistency, we learn a consensus graph filter via linearly combining the view-specific filters:

$$\hat{\mathbf{G}} = \sum_{v=1}^{V} \beta_v \exp(-\eta \mathbf{L}_v), \tag{5}$$

where $\boldsymbol{\beta}$ are weights determined based on the downstream clustering task. This consensus mechanism consolidates high-order topological structures across views, enabling effective noise suppression and yielding more coherent cluster results.

In the LFMKC task, the base partition $\mathbf{H}_p$ can be regarded as a graph signal. If $\mathbf{H}_p$ exhibits a clear clustering structure, it should adhere to the clustering and

manifold assumptions, which state that data within the same class should be close to each other. According to references [4,24], smooth graph signals often follow the clustering and manifold assumptions. Building upon this observation, we apply the above-mentioned consensus low-pass graph filter on the signals $\mathbf{H}_p$ to obtain a more cluster-friendly embedding $\mathcal{F}_L(\mathbf{H}_p)$ as follows:

$$\mathcal{F}_L(\mathbf{H}_p) = \hat{\mathbf{G}}\mathbf{H}_p = \left(\sum_{v=1}^{V} \beta_v \exp(-\eta \mathbf{L}_v)\right) \mathbf{H}_p. \tag{6}$$

The filtered embeddings $\{\mathcal{F}_L(\mathbf{H}_p)\}_{p=1}^{V}$ are expected to exhibit enhanced cluster smoothness. To fuse these enhanced representations into the final clustering result, we learn a discrete consensus indicator matrix $\mathbf{Y} \in \{0,1\}^{n\times c}$ and an orthogonal centroid matrix $\mathbf{C}_p \in \mathbb{R}^{d\times c}$ with c disjoint clusters for each partition, by solving the following prototype-based optimization objective function:

$$\begin{aligned} &\min \sum_{p=1}^{V} \alpha_p \Big\| \Big(\sum_{v=1}^{V} \beta_v \exp(-\eta \mathbf{L}_v) \Big) \mathbf{H}_p - \mathbf{Y}\mathbf{C}_p^\top \Big\|_F^2, \\ &\text{s.t. } \mathbf{C}_p^\top \mathbf{C}_p = \mathbf{I},\ \mathbf{Y} \in \text{Ind},\ \sum_{p=1}^{V} \frac{1}{\alpha_p} = 1,\ \boldsymbol{\beta}^\top \mathbf{1} = 1, \alpha_p \geq 0,\ \beta_v \geq 0,\ \forall p, v, \end{aligned} \tag{7}$$

where α_p quantifies the contribution of each partition in guiding the consensus. The orthogonality constraint $\mathbf{C}_p^\top \mathbf{C}_p = \mathbf{I}$ encourages well-separated clusters and facilitates optimization.

3.3 Joint Clustering via Optimal Filtering with Dual-Frequency

While low-pass graph filters are commonly applied to enhance intra-cluster coherence by smoothing node features, they often treat all high-frequency components as noise and arbitrarily remove them, which may obscure critical structural variations. Prior studies [1,12] reveal that high-frequency graph components encode valuable structural discontinuities, which are essential for preserving local discriminability. Thus, over-reliance on low-pass filters may lead to over-smoothing, blurring boundaries and hindering model generalization [25]. Motivated by this, we extend the objective in Eq. (7) by explicitly incorporating high-frequency information to restore fine-grained local structure. To this end, we similarly design a consensus high-pass graph filter by linearly combining the Laplacian matrices from all views and apply it to the signal $\mathbf{H}_p$ as follows [22]:

$$\mathcal{F}_H(\mathbf{H}_p) = \left(\sum_{v=1}^{V} \gamma_v \mathbf{L}_v\right) \mathbf{H}_p, \tag{8}$$

where $\boldsymbol{\gamma}_v$ denotes the combination coefficients about each high-pass filter $\mathbf{L}_v$.

Rather than discarding high-frequency signals as noise, our model utilizes them to compensate for the information loss induced by over-smoothing. The

joint clustering objective integrates both filtered representations:

$$\begin{aligned}
&\min \sum_{p=1}^{V} \alpha_p (\lambda \|(\sum_{v=1}^{V} \beta_v \mathbf{G}_v)\mathbf{H}_p - \mathbf{Y}\mathbf{C}_p^\top\|_F^2 + (1-\lambda)\|(\sum_{v=1}^{V} \gamma_v \mathbf{L}_v)\mathbf{H}_p - \mathbf{Y}\mathbf{C}_p^\top\|_F^2), \\
&\text{s.t.} \mathbf{C}_p^\top \mathbf{C}_p = \mathbf{I},\ \mathbf{Y} \in \text{Ind},\ \sum_{p=1}^{V} \frac{1}{\alpha_p} = 1,\ \boldsymbol{\beta}^\top \mathbf{1} = 1,\ \boldsymbol{\gamma}^\top \mathbf{1} = 1, \\
&\quad \alpha_p \geq 0,\ \beta_v \geq 0,\ \gamma_v \geq 0,\ \forall p, v,
\end{aligned} \tag{9}$$

where $\lambda \in [0, 1]$ controls the trade-off between global smoothness and local discriminability. By jointly optimizing low- and high-frequency responses, $\mathcal{F}_L(\mathbf{H}_p)$ and $\mathcal{F}_H(\mathbf{H}_p)$, the model effectively captures multi-scale structural cues. This dual-frequency optimal filtering design enhances robustness to noise and over-smoothing, enabling the consensus indicator matrix $\mathbf{Y}$ to encode a more expressive and faithful clustering structure across views.

3.4 Optimization

To solve the optimization problem in Eq. (9), we propose a five-step alternating optimization algorithm.

Update $\{\mathbf{C}_p\}_{p=1}^V$ with fixed $\mathbf{Y}$, $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$, and $\boldsymbol{\gamma}$. Optimizing each $\mathbf{C}_p$ is equivalent to solving the following equation:

$$\max_{\mathbf{C}_p} \ \text{tr}(\mathbf{C}_p^\top \mathbf{A}_p) \quad \text{s.t. } \mathbf{C}_p^\top \mathbf{C}_p = \mathbf{I}, \tag{10}$$

where $\mathbf{A}_p = \mathbf{H}_p^\top \left(\lambda \sum_{v=1}^{V} \beta_v \mathbf{G}_v + (1-\lambda) \sum_{v=1}^{V} \gamma_v \mathbf{L}_v\right) \mathbf{Y}$. The SVD of $\mathbf{A}_p$ is expressed as $\mathbf{A}_p = \mathbf{U}_p \boldsymbol{\Lambda}_p \mathbf{V}_p^\top$. It is obtained that $\mathbf{C}_p$ is updated by:

$$\mathbf{C}_p = \mathbf{U}_p \mathbf{V}_p^\top. \tag{11}$$

For the detailed proof, see [8].

Update $\mathbf{Y}$ with fixed $\{\mathbf{C}_p\}_{p=1}^V$, $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$, and $\boldsymbol{\gamma}$. Optimizing $\mathbf{Y}$ is equivalent to solving the following equation:

$$\max_{\mathbf{Y}} \ \text{tr}(\mathbf{Y}^\top \mathbf{F}) \quad \text{s.t. } \mathbf{Y} \in \text{Ind}, \tag{12}$$

where $\mathbf{F} = \sum_{p=1}^{V} \alpha_p \left(\lambda \sum_{v=1}^{V} \beta_v \mathbf{G}_v + (1-\lambda) \sum_{v=1}^{V} \gamma_v \mathbf{L}_v\right) \mathbf{H}_p \mathbf{C}_p$. The optimal solution for Eq. (12) is given by:

$$y_{ij} = \begin{cases} 1, & \text{if } j = \arg\max_{j'} [\mathbf{F}]_{ij'}, \\ 0, & \text{otherwise.} \end{cases} \tag{13}$$

Update $\boldsymbol{\alpha}$ with fixed $\mathbf{Y}$, $\{\mathbf{C_p}\}_{\mathbf{p=1}}^{\mathbf{V}}$, $\boldsymbol{\beta}$, and $\boldsymbol{\gamma}$. Optimizing $\{\alpha_p\}_{p=1}^{V}$ is equivalent to solving the following equation:

$$\min_{\alpha_p} \sum_{p=1}^{V} \alpha_p \zeta_p, \quad \text{s.t. } \sum_{p=1}^{V} \frac{1}{\alpha_p} = 1,\ \alpha_p \geq 0,\ \forall p, \tag{14}$$

where $\zeta_p = \lambda \|(\sum_{v=1}^{V} \beta_v \mathbf{G}_v \mathbf{H}_p - \mathbf{Y}\mathbf{C}_p^\top\|_F^2 + (1-\lambda)\|\sum_{v=1}^{V} \gamma_v \mathbf{L}_v \mathbf{H}_p - \mathbf{Y}\mathbf{C}_p^\top\|_F^2$. Based on Cauchy-Schwarz inequality, α_p is updated by:

$$\alpha_p = \frac{\sum_{p=1}^{V} \sqrt{\zeta_p}}{\sqrt{\zeta_p}}. \tag{15}$$

Update $\boldsymbol{\beta}$ with fixed $\mathbf{Y}$, $\{\mathbf{C_p}\}_{\mathbf{p=1}}^{\mathbf{V}}$, $\boldsymbol{\alpha}$, and $\boldsymbol{\gamma}$. Optimizing $\boldsymbol{\beta}$ is equivalent to solving the following equation:

$$\min_{\boldsymbol{\beta}} \boldsymbol{\beta}^\top \mathbf{M} \boldsymbol{\beta} - 2\boldsymbol{\beta}^\top \mathbf{b}, \quad \text{s.t. } \boldsymbol{\beta}^\top \mathbf{1} = 1,\ \beta_v \geq 0,\ \forall v, \tag{16}$$

where $\mathbf{M} \in \mathbb{R}^{V \times V}$ with entries $M_{ij} = \sum_{p=1}^{V} \alpha_p \operatorname{tr}\left(\mathbf{H}_p^\top \mathbf{G}_i \mathbf{G}_j \mathbf{H}_p\right)$, and $\mathbf{b} \in \mathbb{R}^V$ with entries $b_i = \sum_{p=1}^{V} \alpha_p \operatorname{tr}\left(\mathbf{Y}^\top \mathbf{G}_i \mathbf{H}_p \mathbf{C}_p\right)$. The resulting quadratic programming problem can be efficiently solved using standard optimization solvers.

Update $\boldsymbol{\gamma}$ with fixed $\mathbf{Y}$, $\{\mathbf{C_p}\}_{\mathbf{p=1}}^{\mathbf{V}}$, $\boldsymbol{\alpha}$, and $\boldsymbol{\beta}$. Optimizing $\boldsymbol{\gamma}$ is equivalent to solving the following equation:

$$\min_{\boldsymbol{\gamma}} \boldsymbol{\gamma}^\top \mathbf{B} \boldsymbol{\gamma} - 2\boldsymbol{\gamma}^\top \mathbf{a}, \quad \text{s.t. } \boldsymbol{\gamma}^\top \mathbf{1} = 1,\ \gamma_v \geq 0,\ \forall v, \tag{17}$$

where $\mathbf{B} \in \mathbb{R}^{V \times V}$ with entries $B_{ij} = \sum_{p=1}^{V} \alpha_p \operatorname{tr}\left(\mathbf{H}_p^\top \mathbf{L}_i \mathbf{L}_j \mathbf{H}_p\right)$, and $\mathbf{a} \in \mathbb{R}^V$ with entries $a_i = \sum_{p=1}^{V} \alpha_p \operatorname{tr}\left(\mathbf{Y}^\top \mathbf{L}_i \mathbf{H}_p \mathbf{C}_p\right)$. The solution can be efficiently computed via standard solvers.

Acceleration for Linear Filtering. The primary computational bottleneck during above updates stems from the $\mathcal{O}(n^2)$ cost of directly computing $\mathbf{G}_v\mathbf{H}_p$ and $\mathbf{L}_v\mathbf{H}_p$, which becomes prohibitive for large n. To address this, we exploit the inherent low-rank structure of the bipartite graph $\mathbf{P}_v$, where its thin SVD decomposition $\mathbf{P}_v = \mathbf{Q}_v \boldsymbol{\Sigma}_v \mathbf{N}_v^\top$ enables efficient spectral filtering:

$$\mathbf{G}_v\mathbf{H}_p = \mathbf{Q}_v \exp\left(-\eta(\mathbf{I} - \boldsymbol{\Sigma}_v^2)\right) \mathbf{Q}_v^\top \mathbf{H}_p, \tag{18}$$

$$\mathbf{L}_v\mathbf{H}_p = \mathbf{Q}_v(\mathbf{I} - \boldsymbol{\Sigma}_v^2)\mathbf{Q}_v^\top \mathbf{H}_p \tag{19}$$

where $\mathbf{Q}_v \in \mathbb{R}^{n \times m_v}$, $\boldsymbol{\Sigma}_v \in \mathbb{R}^{m_v \times m_v}$, and $\mathbf{N}_v \in \mathbb{R}^{m_v \times m_v}$ with $n \gg m_v$. By reordering the matrix products from right to left, the filtering complexity reduces from $\mathcal{O}(n^2 d)$ to $\mathcal{O}(n m_v d)$, ensuring linear scalability with respect to the sample

Algorithm 1 Late Fusion Multiple Kernel Clustering Refined via Optimal Linear Graph Filtering (LFMKC-OLF)

Input: Base partitions $\{\mathbf{H}_p\}_{p=1}^V$, number of clusters c, number of anchors $\{m_v\}_{v=1}^V$, number of nearest neighbors k, the parameter η and λ.
1: Construct sample-anchor similarity graphs $\{\mathbf{Z}_v\}_{v=1}^V$ for each partition then compute the column-normalized bipartite graphs $\{\mathbf{P}_v\}_{v=1}^V$ using Eq. (2).
2: Precompute filtered tensor banks in Eq. (20) using Eq. (18) and Eq. (19).
3: Initialize $\mathbf{Y}$ and $\{\mathbf{C}_p\}_{p=1}^V$.
4: Initialize $\alpha_p \Leftarrow V$, $\beta_v \Leftarrow 1/V$ and $\gamma_v \Leftarrow 1/V$, $\forall p, v$.
5: **while** not converged **do**
6: Calculate the filtered embeddings $\mathcal{F}_L(\mathbf{H}_p)$ and $\mathcal{F}_H(\mathbf{H}_p)$ for all p via weighted aggregation over the precomputed tensor banks with current $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$.
7: Update $\{\mathbf{C}_p\}_{p=1}^V$ according to Eq. (11).
8: Update $\mathbf{Y}$ according to Eq. (13).
9: Update $\boldsymbol{\alpha}$ according to Eq. (15).
10: Update $\boldsymbol{\beta}$ by solving Eq. (16).
11: Update $\boldsymbol{\gamma}$ by solving Eq. (17).
12: **end while**
Output: Final clustering assignment $\mathbf{Y}$.

size. Furthermore, all filtered outputs $\mathbf{G}_v\mathbf{H}_p$ and $\mathbf{L}_v\mathbf{H}_p$ are precomputed and cached prior to optimization, forming two filtered tensor banks:

$$\begin{bmatrix} \mathbf{G}_1\mathbf{H}_1 & \cdots & \mathbf{G}_V\mathbf{H}_1 \\ \vdots & \ddots & \vdots \\ \mathbf{G}_1\mathbf{H}_V & \cdots & \mathbf{G}_V\mathbf{H}_V \end{bmatrix} \text{ and } \begin{bmatrix} \mathbf{L}_1\mathbf{H}_1 & \cdots & \mathbf{L}_V\mathbf{H}_1 \\ \vdots & \ddots & \vdots \\ \mathbf{L}_1\mathbf{H}_V & \cdots & \mathbf{L}_V\mathbf{H}_V \end{bmatrix}. \tag{20}$$

During iterative updates, the consensus filtered embeddings $\mathcal{F}_L(\mathbf{H}_p)$ and $\mathcal{F}_H(\mathbf{H}_p)$ are efficiently obtained by weighted aggregation of the precomputed tensors using the current $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$.

We summarize the procedures of LFMKC-OLF algorithm in Algorithm 1.

3.5 Complexity Analysis

The overall computational complexity of the proposed LFMKC-OLF can be decomposed as follows. Anchor generation and bipartite graph construction incur costs of $\mathcal{O}(nd\sum_{v=1}^V m_v)$ and $\mathcal{O}(nk\sum_{v=1}^V m_v)$, respectively. The precomputation of filtered tensor banks in Eq. (20) requires $\mathcal{O}(\sum_{v=1}^V(nm_vd+m_vd+m_v))$. During iterative optimization, updating $\mathbf{C}_p$, $\mathbf{Y}$, $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$, and $\boldsymbol{\gamma}$ involves complexities of $\mathcal{O}(ndc + dc^3)$, $\mathcal{O}(ndcV)$, $\mathcal{O}(ndcV)$, $\mathcal{O}(n(dV^3 + dcV))$, and $\mathcal{O}(n(dV^3 + dcV))$, respectively. Given that $m_v, c, d, V \ll n$, the overall complexity scales linearly as $\mathcal{O}(n)$, ensuring scalability to large-scale datasets.

Table 1: Dataset Summary.

ID	Dataset	Samples	Features	Classes	Kernels
D1	Caltech	2386	48, 40, 254, 1984, 512, 928	20	6
D2	MouseBladder	2746	11829	16	12
D3	Wiki	2866	128, 10	10	2
D4	Zeisel	3005	4401	48	12
D5	Macosko	6418	8608	39	12
D6	One-Year-Testic	7688	9667	10	12
D7	CITECBMC	8617	1703	15	12
D8	TDT2	9394	36771	30	12
D9	Fetal-Pancreas	11983	11673	18	12
D10	MouseRetina	27499	13166	19	12

4 Experiments

4.1 Experimental Setup

We briefly introduce the experimental setup here, including the used datasets, settings, and implementation details.

Datasets. We evaluate our method on 10 benchmark datasets: Caltech, MouseBladder, Wiki, Zeisel, Macosko, One-Year-Testic, CITECBMC, TDT2, Fetal-Pancreas, and MouseRetina. Detailed information about these datasets is provided in Table 1.

Compared Methods. We compare the proposed LFMKC-OLF with twelve state-of-the-art late fusion MKC methods, including AWP [17], LFMVC [21], OPLFMVC [16], ALMVC [30], LFLKA [29], MMLMVC [11], ERMKC [27], sLGm [23], HKLMVC [26], RIWLF [31], GMLKM [5], and CFGFLF [7]. Among these, HKLMVC, GMLKM, and CFGFLF incorporate graph filtering techniques to enhance clustering performance. Moreover, the average Kmeans result on base partitions (AvgH) is reported as a baseline.

Experimental Setting. All kernel matrices used in our experiments are precomputed using carefully designed similarity functions, and each base kernel is centered and normalized. Specifically, for the Caltech and Wiki datasets, a linear kernel matrix is constructed for each view. For the remaining datasets, we construct a diverse set of 12 kernel matrices per dataset, comprising 7 Gaussian kernels, 1 linear kernel, and 4 polynomial kernels, as detailed in [6]. To assess the performance of all methods under varying structural conditions, we further generate local kernels that capture local structural information based on the original kernels described above, following the strategy outlined in [29]. Throughout our experiments, the base partitions are obtained by performing kernel Kmeans on

the constructed kernels and extracting their top-c eigenvectors, which serve as inputs for subsequent late fusion processes.

Implementation Details. All baseline methods are implemented with their recommended hyperparameter settings as reported in the respective original papers. To reduce random bias, all methods are executed 10 times and the average results are taken. The number of clusters for each dataset is set to the ground-truth class count. We evaluate clustering performance using three common metrics: Accuracy (ACC), Adjusted Rand Index (ARI), and Fscore. All experiments are conducted on a desktop equipped with an AMD Ryzen 7 5700G CPU (3.8 GHz) and 64 GB RAM, using MATLAB R2022b. For the proposed LFMKC-OLF, the neighborhood size k is fixed at 5, and the diffusion parameter η is set to 9 across all experiments without tuning. The balance parameter λ is tuned within the range $\{0.1, 0.2, \ldots, 1.0\}$ with a step size of 0.1. For simplicity, the number of anchors across all views is set as $m_1 = m_2 = \cdots = m_V = m$, where m is searched in $\{4c, 6c, 8c, 10c\}$.

4.2 Experimental Results

Tables 2 and 3 present the clustering results under two distinct base partition settings, with best results highlighted in red and second-best in blue. The proposed LFMKC-OLF consistently outperforms all baseline methods across most datasets in terms of ACC, ARI, and Fscore. In particular, on dataset D6, LFMKC-OLF exceeds the second-best method by substantial margins, with 70.43% under the original kernel partition and 25.65% under the local kernel partition in terms of ACC, underscoring its strength in LFMKC scenarios.

Although methods such as AWP, LFMVC, and OPLFMVC are specifically designed for efficient partition fusion, their performance remains limited, primarily due to reliance on static, low-quality base partitions and a lack of capacity to exploit higher-order structural information. In contrast, graph filter-based methods like HKLMVC, GMLKM, and CFGFLF enhance clustering through low-pass filtering, showing competitive performance on several datasets. However, these methods appear to converge toward a performance ceiling, likely attributable to the over-smoothing effect inherent in their filtering strategies. LFMKC-OLF consistently delivers superior performance improvement over other graph filter-based methods and achieves optimal filtering through dual-frequency joint clustering, effectively avoiding over-smoothing. These results confirm the effectiveness and strong applicability of the proposed method in diverse contexts.

4.3 Running Time Comparison and Convergence

We assess the time efficiency of the proposed LFMKC-OLF algorithm in Fig. 1. Experimental results reveal that LFMKC-OLF consistently outperforms most baseline methods in runtime, validating its computational efficiency. Notably, compared with graph filtering-based approaches such as HKLMVC, GMLKM,

Table 2: Clustering results (in %) on original kernel partitions.

Methods	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10
ACC (%)										
AvgH	32.99	40.02	36.37	29.16	42.85	28.90	35.76	35.06	34.01	34.12
AWP	57.54	51.86	50.70	39.33	64.77	30.15	41.50	53.84	41.76	41.97
LFMVC	42.44	51.63	51.10	37.51	60.30	36.20	39.79	46.51	38.68	42.24
OPLFMVC	52.51	52.13	54.46	39.07	65.37	32.71	41.46	49.95	41.94	41.54
ALMVC	41.41	51.51	50.23	36.87	59.26	35.78	39.77	46.28	38.62	41.97
LFLKA	42.13	52.19	51.10	37.39	60.36	36.20	39.74	47.34	38.30	42.59
MMLMVC	40.23	48.92	50.23	37.13	57.62	34.85	38.67	48.30	37.86	39.71
ERMKC	41.78	58.92	54.82	37.70	59.88	42.20	42.62	49.13	43.52	42.63
sLGm	43.30	51.51	51.08	37.14	58.74	36.12	39.43	46.16	39.24	42.58
HKLMVC	60.55	55.87	55.64	45.42	70.67	35.90	42.68	54.72	46.43	46.42
RIWLF	42.73	51.88	51.55	37.75	59.93	36.44	39.60	46.70	39.21	42.32
GMLKM	47.11	50.78	55.34	39.95	59.04	34.78	45.14	40.40	40.88	46.32
CFGFLF	54.17	54.97	55.34	43.12	64.09	33.98	48.26	49.08	48.76	45.09
LFMKC-OLF	73.26	76.47	55.79	51.41	77.45	71.92	65.85	78.90	64.78	75.06
ARI (%)										
AvgH	20.23	26.28	22.38	14.59	34.16	9.51	20.47	16.88	15.30	16.64
AWP	51.47	38.44	34.11	22.39	54.16	13.38	30.43	35.71	23.46	24.79
LFMVC	30.96	39.35	35.57	21.77	49.35	18.89	29.24	31.55	22.63	26.32
OPLFMVC	42.16	38.68	42.97	22.26	55.67	15.15	30.11	33.37	23.38	25.00
ALMVC	30.46	39.38	34.71	21.54	49.09	18.53	29.24	31.38	22.60	26.30
LFLKA	31.24	39.74	35.57	21.34	50.16	18.90	29.07	32.14	22.46	26.82
MMLMVC	29.64	36.21	34.71	20.91	45.68	16.91	29.01	32.53	22.33	23.45
ERMKC	30.39	45.01	42.35	21.89	49.39	21.09	32.58	37.74	27.61	26.59
sLGm	31.21	39.50	35.55	21.64	48.00	18.96	29.14	30.94	22.87	26.33
HKLMVC	50.74	42.38	44.04	30.81	62.94	18.53	31.47	40.57	27.94	30.22
RIWLF	30.35	39.79	36.55	21.83	49.41	19.30	29.16	31.29	22.76	26.24
GMLKM	33.88	38.01	43.38	24.88	47.76	17.54	33.41	20.43	25.48	31.31
CFGFLF	45.50	40.24	43.54	27.12	53.95	17.47	36.06	32.61	30.29	32.10
LFMKC-OLF	73.05	65.70	40.34	45.46	75.79	63.51	51.90	72.82	39.03	82.56
Fscore (%)										
AvgH	26.63	35.07	30.52	18.82	36.67	27.04	34.07	26.12	27.78	27.71
AWP	57.01	44.11	41.09	25.13	56.14	27.63	37.46	39.57	31.06	31.49
LFMVC	36.64	44.91	42.33	24.40	51.28	31.96	36.26	35.40	29.55	32.60
OPLFMVC	48.00	44.34	49.04	24.95	57.60	28.67	37.13	37.16	31.14	31.63
ALMVC	36.17	44.93	41.55	24.17	51.02	31.64	36.26	35.22	29.52	32.59
LFLKA	36.97	45.30	42.33	23.97	52.08	31.96	36.09	35.96	29.37	33.14
MMLMVC	35.36	41.98	41.55	23.56	47.72	30.15	35.95	36.35	29.25	29.92
ERMKC	36.10	50.35	48.47	24.59	51.53	33.92	40.39	41.51	34.34	32.89
sLGm	36.85	45.05	42.31	24.28	49.97	32.02	36.17	34.80	29.79	32.74
HKLMVC	57.83	47.94	50.07	34.00	64.70	32.62	38.51	44.44	35.72	36.80
RIWLF	36.07	45.33	43.20	24.46	51.33	32.35	36.15	35.14	29.69	32.52
GMLKM	40.23	43.94	49.42	27.58	49.86	31.16	40.72	27.04	32.81	37.96
CFGFLF	51.59	46.23	49.61	30.02	55.98	31.65	43.01	38.42	38.51	38.78
LFMKC-OLF	78.16	70.51	47.71	49.59	77.26	77.33	61.25	75.65	52.39	85.94

and CFGFLF, LFMKC-OLF exhibits superior scalability and efficiency on large-scale datasets. Although CFGFLF also achieves linear filtering, its high per-iteration complexity leads to substantially longer overall runtime. In contrast, the lightweight design of LFMKC-OLF ensures low per-iteration overhead, achieving a favorable trade-off between computational cost and clustering quality.

Convergence behavior on Caltech, MouseBladder, Wiki, and Macosko datasets is illustrated in Fig. 2, with similar trends on other datasets omitted for brevity. The objective value decreases monotonically and converges within 15 iterations, accompanied by steady improvements in clustering performance until stabilization. These results underscore the algorithm's fast convergence and effectiveness.

Table 3: Clustering results (in %) on local kernel partitions.

Methods	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10
ACC (%)										
AvgH	33.68	49.21	36.21	30.23	34.79	32.43	57.67	60.55	29.87	32.47
AWP	47.90	58.78	45.43	36.14	35.87	44.25	65.70	78.77	40.44	41.27
LFMVC	43.98	50.29	51.34	30.53	34.82	32.18	58.05	58.69	30.48	31.93
OPLFMVC	50.44	59.13	55.22	34.03	37.22	44.18	65.63	79.21	40.40	41.41
ALMVC	42.28	49.03	50.64	30.43	34.49	31.89	56.92	58.69	30.15	31.76
LFLKA	43.37	50.24	51.35	30.52	35.26	32.09	57.77	59.55	30.48	31.92
MMLMVC	41.92	49.09	50.63	30.80	34.73	32.63	57.28	58.42	30.60	31.87
ERMKC	50.39	53.47	53.33	31.89	35.69	35.16	61.80	64.40	32.87	36.24
sLGm	43.32	49.66	51.10	30.74	34.46	32.11	57.91	58.84	30.62	31.96
HKLMVC	52.42	56.31	55.25	34.91	37.84	44.36	65.97	78.29	40.73	41.48
RIWLF	44.20	50.17	51.12	30.60	34.81	47.20	65.83	60.79	36.84	45.22
GMLKM	45.54	49.25	54.62	34.18	35.60	41.47	58.82	61.63	31.87	35.92
CFGFLF	43.40	54.88	55.20	39.58	39.79	52.40	67.09	84.48	43.54	53.38
LFMKC-OLF	63.24	73.71	63.08	45.99	46.32	65.84	69.59	85.05	52.92	61.35
ARI (%)										
AvgH	20.84	37.69	22.63	20.37	28.38	17.88	47.51	49.88	16.05	21.04
AWP	32.54	42.37	26.39	27.30	30.55	31.03	54.53	70.16	21.61	40.08
LFMVC	30.06	38.78	35.41	20.29	28.26	18.04	47.62	47.30	16.31	22.57
OPLFMVC	36.28	43.41	43.54	25.04	31.05	30.96	54.57	68.13	21.60	40.01
ALMVC	29.58	37.99	34.99	20.37	28.13	18.11	47.07	47.30	16.13	22.02
LFLKA	29.78	38.83	35.49	20.50	28.82	18.21	47.63	48.20	16.31	22.56
MMLMVC	27.65	37.42	34.97	20.42	28.24	18.16	47.32	47.89	16.39	22.52
ERMKC	38.89	41.71	38.87	21.73	29.61	19.10	50.60	51.16	17.61	25.03
sLGm	30.56	38.67	35.09	20.37	27.90	18.11	47.64	47.17	16.47	22.70
HKLMVC	38.74	41.75	43.20	25.74	30.91	30.74	54.53	68.52	21.70	40.15
RIWLF	29.37	38.41	34.98	20.51	28.90	15.49	54.47	49.54	21.83	28.08
GMLKM	27.34	37.34	39.57	24.44	26.70	24.62	46.03	43.21	16.47	25.72
CFGFLF	25.22	38.99	42.81	29.92	30.27	3.23	54.62	66.54	22.58	52.20
LFMKC-OLF	49.65	60.23	49.96	42.66	37.15	51.00	58.65	78.31	31.14	66.83
Fscore (%)										
AvgH	27.91	43.57	30.97	23.31	31.24	32.03	56.67	53.83	23.51	29.93
AWP	40.03	48.99	34.21	30.70	34.02	49.29	63.03	72.92	31.65	48.25
LFMVC	36.22	44.58	42.26	23.25	31.07	31.95	56.75	50.88	23.79	29.61
OPLFMVC	43.32	49.80	49.67	28.19	34.48	49.25	63.06	70.97	31.64	48.19
ALMVC	35.76	43.81	41.84	23.31	30.98	32.16	56.23	50.88	23.58	29.09
LFLKA	36.03	44.61	42.32	23.42	31.64	32.18	56.72	51.76	23.80	29.60
MMLMVC	33.94	43.35	41.83	23.38	31.07	32.06	56.47	51.45	23.91	29.63
ERMKC	45.32	47.48	45.41	24.71	32.52	33.06	59.44	54.73	25.11	32.09
sLGm	36.75	44.42	41.98	23.30	30.70	32.04	56.77	50.79	23.99	29.72
HKLMVC	46.23	48.16	49.45	29.07	34.43	49.13	63.05	71.41	31.80	48.34
RIWLF	35.51	44.19	41.86	23.43	31.70	44.57	62.86	52.98	29.90	45.71
GMLKM	36.65	43.56	46.58	28.05	30.19	41.79	55.39	48.56	25.51	34.82
CFGFLF	34.75	46.50	49.14	34.16	34.64	47.74	63.21	70.88	34.97	61.34
LFMKC-OLF	58.87	65.92	56.17	47.19	41.57	68.44	66.64	80.54	43.06	72.93

4.4 Parameter Analysis

We evaluate the sensitivity of LFMKC-OLF to two key parameters: the number of anchors $m \in \{4c, 6c, 8c, 10c\}$, where c is the number of clusters, and the trade-off parameter $\lambda \in \{0.1, 0.2, \ldots, 1.0\}$. The performance trends under different parameter combinations are illustrated in Fig. 3. We observe that performance remains relatively stable as the number of anchors m varies. Regarding the trade-off parameter λ, optimal performance is typically achieved within the range $[0.6, 0.8]$, while larger values tend to over-smoothing and performance degradation, suggesting that the smoothing component plays a dominant role. Notably, on datasets such as Caltech and Wiki, better performance is observed with smaller λ values in the range $[0.2, 0.5]$, highlighting the effectiveness of high-

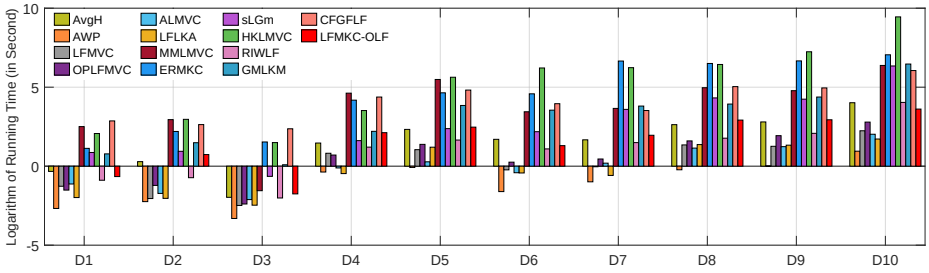
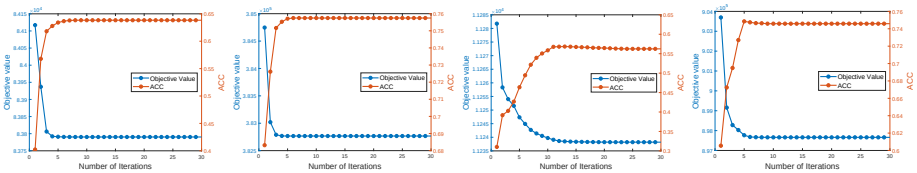


Fig. 1: Comparison of LFMKC-OLF and baseline methods in terms of relative base-10 logarithmic runtime on ten benchmark datasets under the original kernel partitions.



(a) Caltech (b) MouseBladder (c) Wiki (d) Macosko

Fig. 2: The objective value and clustering performance of LFMKC-OLF vary with increasing iterations on Caltech, MouseBladder, Wiki, and Macosko datasets.

pass filtering in preserving fine-grained details, alleviating over-smoothing and further enhancing representations to improve clustering performance.

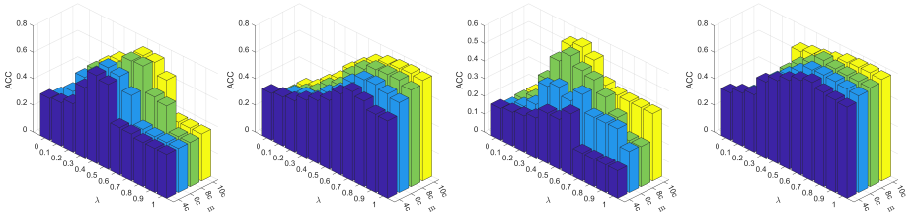
4.5 Ablation Study

To assess the contribution of each component in the LFMKC-OLF objective, we perform an ablation study comparing the full model against two simplified variants: (1) a baseline defined by Eq. (7) without the consensus low-pass filter $\hat{\mathbf{G}}$, denoted as Eq. (7) w/o $\hat{\mathbf{G}}$; (2) the low-pass-only model given by Eq. (7).

The results, summarized in Table 4 with the best values highlighted in red, demonstrate that LFMKC-OLF consistently outperforms both variants across all datasets. This clearly verifies the effectiveness of both low-pass filtering and high-frequency enhancement in improving clustering performance. Notably, the performance gap between Eq. (7) w/o $\hat{\mathbf{G}}$ and Eq. (7) highlights the importance of incorporating the low-pass filter, which enhances global structural consistency and suppresses noise in base partitions. Furthermore, the superiority of LFMKC-OLF over Eq. (7) confirms the necessity of integrating high-frequency signals to mitigate over-smoothing and retain essential local discriminative information.

5 Conclusion

In this work, we propose a novel late fusion multiple kernel clustering framework refined via optimal linear graph filtering (LFMKC-OLF), which effectively



(a) Caltech (b) MouseBladder (c) Wiki (d) Macosko

Fig. 3: The parameter sensitivity experiments on Caltech, MouseBladder, Wiki, and Macosko datasets under the original kernel partitions.

Table 4: Ablation study: ACC (in %) on different datasets.

Methods	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10
Original Kernel Partitions										
Eq. (7) w/o $\hat{\mathbf{G}}$	57.46	52.55	49.93	37.34	64.82	31.22	45.00	53.23	43.97	39.47
Eq. (7)	33.45	75.56	37.89	49.05	74.21	71.49	56.37	78.06	64.75	74.64
LFMKC-OLF	73.26	76.47	55.79	51.41	77.45	71.92	65.85	78.90	64.78	75.06
Local Kernel Partitions										
Eq. (7) w/o $\hat{\mathbf{G}}$	51.05	59.18	45.43	32.95	34.76	44.35	65.70	80.75	45.48	41.07
Eq. (7)	62.78	68.90	53.80	38.87	43.02	65.83	66.65	82.90	48.92	53.74
LFMKC-OLF	63.24	73.71	63.08	45.99	46.32	65.84	69.59	85.05	52.92	61.35

addresses the limitations of low-quality base partitions while maintaining linear computational complexity. By constructing bipartite graphs and deriving high-order low-pass and high-pass filters through heat kernel diffusion and Laplacian operators, the proposed method captures both global and fine-grained structural information. A consensus filtering mechanism is further introduced by optimally combining view-specific filters, enabling robust multi-scale representation learning. The joint clustering formulation, which integrates both smoothed and detail-enhanced partitions, effectively mitigates over-smoothing and improves clustering fidelity. Extensive experiments across multiple datasets and diverse base partition settings consistently demonstrate the superiority and robustness of LFMKC-OLF. Future work will explore more advanced graph filtering paradigms and extend the framework to broader real-world applications.

Acknowledgments. This work was supported by the National Natural Science Foundation of China (62376146), the Shanxi Province Central Guidance for Local Science and Technology Development Special Project (YDZJSX20231D003).

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Bo, D., Wang, X., Shi, C., Shen, H.: Beyond low-frequency information in graph convolutional networks. In: Proceedings of the AAAI conference on artificial intelligence. vol. 35, pp. 3950–3957 (2021)
2. Chen, Y., Du, L., Zhou, P., Duan, L., Qian, Y.: Multiple kernel clustering with local kernel reconstruction and global heat diffusion. Information Fusion **105**, 102219 (2024)
3. Chung, F.: The heat kernel as the pagerank of a graph. National Academy of Sciences **104**(50), 19735–19740 (2007)
4. Dong, X., Thanou, D., Frossard, P., Vandergheynst, P.: Learning laplacian matrix in smooth graph signal representations. IEEE Transactions on Signal Processing **64**(23), 6160–6173 (2016)
5. Du, L., Jiang, H., Li, X., Guo, Y., Chen, Y., Li, F., Zhou, P., Qian, Y.: Sharper error bounds in late fusion multi-view clustering with eigenvalue proportion optimization. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 39, pp. 16381–16388 (2025)
6. Du, L., Zhou, P., Shi, L., Wang, H., Fan, M., Wang, W., Shen, Y.D.: Robust multiple kernel k-means using ℓ_{21}-norm. In: International Joint Conference on Artificial Intelligence. p. 3476–3482 (2015)
7. Guo, Y., Jiang, H., Chen, Y., Du, L.: A scalable consensus fast graph filtering approach for late fusion multi-view clustering. Signal Processing **237**, 110074 (2025)
8. Huang, J., Nie, F., Huang, H., Ding, C.: Robust manifold nonnegative matrix factorization. ACM Transactions on Knowledge Discovery from Data **8**(3), 1–21 (2014)
9. Isufi, E., Gama, F., Shuman, D.I., Segarra, S.: Graph filters for signal processing and machine learning on graphs. IEEE Transactions on Signal Processing **72**, 4745–4781 (2024)
10. Kang, Z., Xie, X., Li, B., Pan, E.: Cdc: a simple framework for complex data clustering. IEEE Transactions on Neural Networks and Learning Systems (2024)
11. Li, M., Liu, X., Zhang, Y., Liang, W.: Late fusion multiview clustering via min-max optimization. IEEE Transactions on Neural Networks and Learning Systems **35**(7), 9417–9427 (2024)
12. Li, S., Kim, D., Wang, Q.: Beyond low-pass filters: Adaptive feature propagation on graphs. In: Joint European Conference on Machine Learning and Knowledge Discovery in Databases. pp. 450–465. Springer (2021)
13. Li, Z., Tang, C., Zheng, X., Liu, X., Zhang, W., Zhu, E.: High-order correlation preserved incomplete multi-view subspace clustering. IEEE Transactions on Image Processing **31**, 2067–2080 (2022)
14. Lin, Z., Kang, Z.: Graph filter-based multi-view attributed graph clustering. In: International Joint Conference on Artificial Intelligence. pp. 2723–2729 (2021)
15. Liu, X.: Simplemkkm: Simple multiple kernel k-means. IEEE Transactions on Pattern Analysis and Machine Intelligence **45**(4), 5174–5186 (2023)
16. Liu, X., Liu, L., Liao, Q., Wang, S., Zhang, Y., Tu, W., Tang, C., Liu, J., Zhu, E.: One pass late fusion multi-view clustering. In: International Conference on Machine Learning. pp. 6850–6859 (2021)
17. Nie, F., Tian, L., Li, X.: Multiview clustering via adaptively weighted procrustes. In: ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. p. 2022–2030 (2018)

18. Nie, F., Wang, X., Jordan, M.I., Huang, H.: The constrained laplacian rank algorithm for graph-based clustering. In: Proceedings of the AAAI Conference on Artificial Intelligence. p. 1969–1976 (2016)
19. Ren, Z., Sun, Q.: Simultaneous global and local graph structure preserving for multiple kernel clustering. IEEE transactions on neural networks and learning systems **32**(5), 1839–1851 (2020)
20. Ren, Z., Sun, Q., Wei, D.: Multiple kernel clustering with kernel k-means coupled graph tensor learning. In: Proceedings of the AAAI conference on artificial intelligence. vol. 35, pp. 9411–9418 (2021)
21. Wang, S., Liu, X., Zhu, E., Tang, C., Liu, J., Hu, J., Xia, J., Yin, J.: Multi-view clustering via late fusion alignment maximization. In: International Joint Conference on Artificial Intelligence. p. 3778–3784 (2019)
22. Wen, Z., Ling, Y., Ren, Y., Wu, T., Chen, J., Pu, X., Hao, Z., He, L.: Homophily-related: Adaptive hybrid graph filter for multi-view graph clustering. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 38, pp. 15841–15849 (2024)
23. Wu, X., Huang, C., Liu, X., Zhou, F., Ren, Z.: Multiple kernel clustering with shifted laplacian on grassmann manifold. In: ACM International Conference on Multimedia. pp. 2448–2456 (2024)
24. Wu, Z., Pan, S., Long, G., Jiang, J., Zhang, C.: Beyond low-pass filtering: Graph convolutional networks with automatic filtering. IEEE Transactions on Knowledge and Data Engineering **35**(7), 6687–6697 (2022)
25. Xie, X., Chen, W., Kang, Z., Peng, C.: Contrastive graph clustering with adaptive filter. Expert Systems with Applications **219**, 119645 (2023)
26. Yang, G., Zou, J., Chen, Y., Du, L., Zhou, P.: Heat kernel diffusion for enhanced late fusion multi-view clustering. IEEE Signal Processing Letters **31**, 2310–2314 (2024)
27. Yang, W., Tang, C., Zheng, X., Zhu, X., Liu, X.: Eigenvalue ratio inspired partition learning and fusion for multiple kernel clustering. IEEE Transactions on Knowledge and Data Engineering (01), 1–13 (2024)
28. Zhang, P., Wang, S., Li, L., Zhang, C., Liu, X., Zhu, E., Liu, Z., Zhou, L., Luo, L.: Let the data choose: Flexible and diverse anchor graph fusion for scalable multi-view clustering. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 37, pp. 11262–11269 (2023)
29. Zhang, T., Liu, X., Gong, L., Wang, S., Niu, X., Shen, L.: Late fusion multiple kernel clustering with local kernel alignment maximization. IEEE Transactions on Multimedia **25**, 993–1007 (2023)
30. Zhang, T., Liu, X., Zhu, E., Zhou, S., Dong, Z.: Efficient anchor learning-based multi-view clustering – a late fusion method. In: ACM International Conference on Multimedia. p. 3685–3693 (2022)
31. Zhang, Y., Tian, F., Ma, C., Li, M., Yang, H., Liu, Z., Zhu, E., Liu, X.: Regularized instance weighting multiview clustering via late fusion alignment. IEEE Transactions on Neural Networks and Learning Systems pp. 1–13 (2024)
32. Zhao, B., Kwok, J.T., Zhang, C.: Multiple kernel clustering. In: Proceedings of the 2009 SIAM international conference on data mining. pp. 638–649 (2009)
33. Zhou, P., Du, L.: Learnable graph filter for multi-view clustering. In: ACM International Conference on Multimedia. p. 3089–3098 (2023)
34. Zhou, P., Du, L., Li, X.: Adaptive consensus clustering for multiple k-means via base results refining. IEEE Transactions on Knowledge and Data Engineering **35**(10), 10251–10264 (2023)