Joint Graph Matching and Clustering with Majorization-minimization Optimization and Edge-enhanced Unsupervised Learning

Anonymous CVPR submission

Paper ID ****

Abstract

Given a batch of graphs that often do not share the same or similar structure in real-world scenarios, one challenging and important task is to cluster the graphs into groups and further find the node correspondences among the graphs within each group. In fact, matching and clustering are two interleaved tasks and a joint approach is attractive yet rarely studied. This work aims to push forward the graph matching models to this more general setting: mixture graph matching and clustering. Specifically, a theoretically convergent Majorization-minimization style algorithm namely M3C is devised for joint matching and clustering. Based on M3C, an unsupervised learning model UM3C is further developed which is equipped with our devised edge-wise affinity learning and pseudo label selection techniques. Experimental results on public benchmarks show that our method notably surpasses the state-of-the-art methods in both accuracy and efficiency, where our unsupervised model even exceeds supervised methods.

1. Introduction

Graph matching (GM) has been a standing problem and it has found wide applications in vision, such as image registration [28], recognition [7, 8], stereo [12], 3D shape matching [2, 25], and structure from motion [29]. In general, the problem refers to finding the node correspondence between two or multiple graphs by maximizing the resulting affinity score between the matched nodes and edges:

$$\begin{aligned} &\mathbf{X} = \arg\max_{\mathbf{X}} \mathrm{vec}(\mathbf{X})^\top \mathbf{K} \mathrm{vec}(\mathbf{X}) \\ &s.t.\ \ \mathbf{X} \in \{0,1\}^{n_1 \times n_2}, \mathbf{X}\mathbf{1}_{n_2} \leq \mathbf{1}_{n_1}, \mathbf{X}^\top \mathbf{1}_{n_1} \leq \mathbf{1}_{n_2} \end{aligned} \tag{1}$$

where $\mathbf{X}$ is a permutation matrix encoding node-to-node correspondence, and $\mathrm{vec}(\mathbf{X})$ is its column-vectorized version. $\mathbf{1}_n$ represents column vector of length n whose elements all equal to 1. $\mathbf{K} \in \mathbb{R}^{n_1 n_2 \times n_1 n_2}$ is the affinity matrix. A popular extension for two-graph matching above is multiple graph matching (MGM) [17, 37, 38] where cycle consistency [37, 38] among pairwise matching results are often enforced. Both GM and MGM problems are NP-hard with approximate algorithms proposed over the decades.

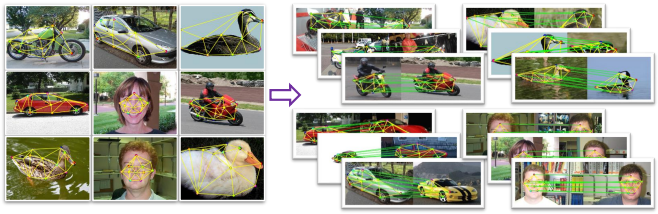

Figure 1. Mixture graph matching and clustering. **Input:** mixture images with different classes where key point coordinates are given and graphs are built with Delaunay triangulation [6]; **Output:** clustered graphs with pairwise mapping in each cluster.

One rarely rethinking assumption in GM literature is that graphs for matching refer to the same category, which requires labeling. On the one hand, labeling is difficult and expensive. It even requires domain knowledge in specific fields, *e.g.* for molecular design or drug discovery, labeled datasets are usually limited in size where large amounts of data lack of detailed annotation and different types of molecules are often mixed up. On the other hand, some real-world situations are naturally of mixture modes, *e.g.* in traffic tracing, people, bicycles, and cars will appear simultaneously in one frame and need to be traced together. Therefore, matching with mixtures of graphs is a practical problem while is still in its early stage. In this paper, we call it Mixture Graph Matching and Clustering (MGMC), which aims to align graph-structured data and divide them into different clusters at the same time, as shown in Fig. 1.

Two recent works [33, 35] have explored the same problem setting i.e. for mixture graph matching and clustering. DPMC [35] is proposed as a learning-free optimization solver which constructs a maximum spanning tree and applies the decay ratio based on affinity score to adjust the weight for different graph pairs. One major drawback is that the tree is an excessive restriction as it leaves out most pairwise results. DPMC heavily relies on the correctness of tree

structure, which tends to fall into a sub-optimum and leads to high performance variance. GANN [33] proposes an unsupervised learning pipeline where learning-free solvers are used to predict pseudo labels for GM training. However, GANN is vulnerable against noise, *e.g*. the matching accuracy drops drastically with 2 outliers, and the model hardly works when outliers further increase. It attributes to the weak solver who focuses too much on node affinity and lacks guidance of learned edge affinity. Moreover, GANN selects all the pairs within the inferred cluster to compute loss, some of whose accuracy can be relatively low.

Specifically, in this paper we first propose a learning-free solver **M3C**: **M**ajorization **M**inimization **M**atching and **C**lustering. It iteratively updates the pairwise matching and cluster indicator to reach the optimal solutions of both. To avoid rigid structure restriction, the constraints of cluster are loosed. The relaxed cluster indicator is more flexible and takes a broad view of overall matching information. To extract better feature from source data **UM3C** is designed fusing **M3C** and the unsupervised learning pipeline [33]. In the training phase, we select the pseudo label pair with higher quality under the guidance of relaxed cluster indicator. Edge-wise affinity construction, as well as affinity loss is added to conform the learning-free solver, which dramatically improves both matching accuracy and robustness of **UM3C**. **The main contributions are:**

1) For the setting of matching graphs from a mixture of modes with clustering , we present a learning-free solver **M3C**. It relaxes the constraints of cluster indicator, which allows a free structure between multiple graphs and considers more pairwise matching results.

2) We further embed our method **M3C** into an unsupervised learning pipeline, so-called **UM3C**. Edge-wise affinity learning and pair selection of pseudo labels are introduced to improve quality and robustness.

3) Our method **M3C** and **UM3C** outperform all kinds of SOTA on learning-free and learning experiments on WillowObject dataset, respectively. Both matching accuracy and cluster quality are improved, where unsupervised model **UM3C** even outperforms supervised models like BBGM, NGM on mixture graph matching and clustering.

2. Related Work

Multi-graph matching. The problem has attracted increasing attention over the past few years. One line of research [3,14,15,18,20,24,41] adopt the matrix factorization paradigm to enforce cycle consistency over the two-graph matching solutions. These works mostly reconstruct initial (noisy) matching to enforce cycle consistency as well as sparsity regularization and node-wise affinity supervision. Another line of studies [17,35,37,38] explicitly introduces the so-called supergraph such that the two-graph matchings are iteratively updated by combining the affinity and consistency along the path in the supergraph for maximization. In contrast, the matrix factorization based techniques work in a post-processing way given initial two-graph matchings, and the affinity information is often not considered. As a result, the supergraph based approaches often outperform in terms of accuracy and efficiency. In this paper, we adopt the supergraph to establish our matching and clustering framework, which naturally lends itself to a decent MM algorithm.

Mixture graph matching and clustering. There are two classic methods in this setting: i) first computing the graph to graph similarity via some means *e.g*. two-graph matching [5], then spectral clustering can be performed to obtain the final clusters; ii) first embedding the whole input graph for matching into an embedding vector [1], and then either k-means like algorithm or spectral clustering can be used for final clustering. In this paper, we resort to the first thread, while we manage to establish a joint matching and clustering pipeline based on the MM framework. This pipeline makes our method significantly different from the two-stage baselines. Perhaps the most related work refers to [35] which addresses a similar problem to ours. Its key idea is to selectively perform matching for those graph pairs that are more likely to be within the same cluster and discourage the matching in separate clusters. However, it imposes hard constraints to construct a maximum spanning tree on the supergraph, with a limitation of only one path between each pair of graphs, leading to sensitivity to the noise in supergraph construction, especially for large-scale inputs. Detailed discussion will be given in the method part.

Learning model for graph matching. By the seminal work [40], deep learning was introduced into GM. GMN [40] applies VGG-16 as a front-end feature extractor and thus the learned affinity's quality exceeds the hand-craft one. PCA [31, 32] adds GCN to the refined node feature and proposed permutation loss for graph matching training. BBGM [27] first introduce a black-box backbone to achieve the combination with non-differentiable solver and deep learning module in an end-to-end model. It adopts Spline CNN to improve the node feature while edge affinity is also learned. NGM [34, 36] addresses the most general Lawler's QAP form with the learnable graph matching solver. DLGM [39], based on BBGM, predicts consistent topology of graphs utilizing both deterministic and generative modelsy. All these works focus on supervised learning, where GANN [33] proposes an unsupervised pipeline with pseudo labels predicted by the devised learning-free solver.

In this paper, we will show a more principled optimization scheme to solve the joint matching and clustering task, under a majorization-minimization framework, and meanwhile we aim to develop a more effective learning pipeline by exploring the edges and pseudo labels for unsupervised learning, whereby no ground truth label is needed.

3. Preliminaries

We first introduce some concepts and notations.

Definition 1 (**Miscellaneous Notations**). *Let* $\mathbb{X} = \{\mathbf{X}_{ij}\}_{i,j=1}^{N,N}$ *denote all the matching results of multi-graph matching, where* $\mathbf{X}_{ij} \in \{0,1\}^{n_i \times n_j}$ *denotes a specific pairwise matching result. Let* $\mathbb{K} = \{\mathbf{K}_{ij}\}_{i,j=1}^{N,N}$ *denote all the affinity matrices where* $\mathbf{K}_{ij} \in \mathbb{R}^{n_i n_j \times n_i n_j}$ *denotes the affinity matrix between* $\mathcal{G}_1$ *and* $\mathcal{G}_j$*. Note* N *denotes the number of graph and* n_i *denotes number of node in* $\mathcal{G}_i$*.*

Definition 2 (**Supergraph**). *Supergraph [37, 38] is a common protocol to describe the multi-graph matching. Let supergraph* $\mathcal{S} = \{\mathcal{V}, \mathcal{E}, \mathbf{A}\}$ *whose vertices are graphs* $\mathcal{V} = \{\mathcal{G}_1, \dots, \mathcal{G}_N\}$ *and edges are weighted by affinity score of pairwise matching* $\mathbf{X}_{ij}$*.* $\mathbf{A}$ *denotes the adjacency of supergraph. Moreover, the value of the path from* $\mathcal{G}_1$ *to* $\mathcal{G}_k$ *(*$\{e_{v_1v_2}, \dots, e_{v_{k-1}v_k}\}$*) is the affinity score of matching composition* $\mathbf{X}_{v_1v_k} = \mathbf{X}_{v_1v_2}\mathbf{X}_{v_2v_3}\dots\mathbf{X}_{v_{k-1}v_k}$*.*

Definition 3 (**Matching composition**). *As first applied in [17, 37], matching composition is a combination of pairwise matching results like:* $\mathbf{X}_{\mathcal{G}_1\mathcal{G}_k} = \mathbf{X}_{\mathcal{G}_1\mathcal{G}_2}\mathbf{X}_{\mathcal{G}_2\mathcal{G}_3}\dots\mathbf{X}_{\mathcal{G}_{k-1}\mathcal{G}_k}$*. We further define matching composition space of* $\mathcal{G}_i$ *and* $\mathcal{G}_j$ *to represent all the possible composition between them:*

$$\mathbf{P}_{\mathbf{A}}(i,j) = \{\mathbf{X}_{\mathcal{G}_i,\mathcal{G}_1}\dots\mathbf{X}_{\mathcal{G}_k,\mathcal{G}_j} | \mathbf{A}_{i1} = \dots = \mathbf{A}_{k,j} = 1\} \tag{2}$$

where $\mathbf{A}_{ij}$ *denotes whether* $\mathbf{X}_{ij}$ *exists. Note each matching composition denotes a path on supergraph, and* $\mathbf{P}_{\mathbf{A}}(i,j)$ *denotes all the paths from* $\mathcal{G}_i$ *to* $\mathcal{G}_j$ *based on adjacency* $\mathbf{A}$*.*

Definition 4 (**Cluster indicator**). *Let* $\mathbf{C} \in \{0,1\}^{N\times N}$ *to be the cluster indicator as:*

$$\mathbf{C}_{ij} = \begin{cases} 1 & \mathcal{G}_i, \mathcal{G}_j \text{ are of the same class.} \\ 0 & \text{Otherwise.} \end{cases} \tag{3}$$

The transitive relation $\mathbf{C}_{ij}\mathbf{C}_{jk} \le \mathbf{C}_{ik}$ *is the sufficient and necessary condition for* $\mathbf{C}$ *to be a strict cluster division. Proof will be given in supplementary material. The number of cluster can be represented by the number of* ***strongly connected component*** *of* $\mathbf{C}$*, which is denoted as* $\texttt{SCC}(\mathbf{C})$*.*

4. Methodology

In this section, we will propose two methods: learning-free solver M3C and unsupervised learning model UM3C. In Sec. 4.1 we first explain the core idea of both methods: relaxation of cluster constraints. In Sec. 4.2 and Sec. 4.3, M3C and UM3C are introduced in detail respectively.

4.1. Relaxation on Cluster Indicator

Essentially, MGMC can be formulated as a joint optimization problem. Matching results maximize the pair similarity to facilitate the clustering process, while the cluster indicator guides matching optimization in turn.

Given the set of pairwise affinity matrix $\mathbb{K}$ and number of clusters N_c, the optimization function can be written as:

$$\begin{aligned} \mathbb{X}, \mathbf{C} = &\arg\max_{\mathbb{X},\mathbf{C}} \sum_{ij} \mathbf{C}_{ij}\text{vec}(\mathbf{X}_{ij})^\top \mathbf{K}_{ij}\text{vec}(\mathbf{X}_{ij}) \\ s.t.\ \ &\mathbf{X}_{ij}\mathbf{1}_{n_j} \le \mathbf{1}_{n_i}, \mathbf{X}_{ij}^\top \mathbf{1}_{n_i} \le \mathbf{1}_{n_j}, \\ &\mathbf{X}_{ik}\mathbf{X}_{kj} \le \mathbf{X}_{ij}, \forall \mathbf{C}_{ik} = \mathbf{C}_{kj} = \mathbf{C}_{ij} = 1 \\ &\mathbf{C}_{ik}\mathbf{C}_{kj} \le \mathbf{C}_{ij}, \forall i,j,k,\ \texttt{SCC}(\mathbf{C}) = N_c \\ &\mathbf{X}_{ij} \in \{0,1\}^{n_i\times n_j}, \mathbf{C}_{ij} \in \{0,1\}. \end{aligned} \tag{4}$$

The first line of constraints guarantees $\mathbf{X}_{ij}$ is a (partial) permutation matrix, and the second line enforces the cycle consistency within each cluster. The third line requires the indicator $\mathbf{C}$ to be a strict cluster division with N_c clusters.

Mutually updating both matching $\mathbb{X}$ and clustering $\mathbf{C}$ is difficult, but optimizing one of them separately has been wildly explored. Intuitively, Majorization-Minimization [16] can be adopted to solve MGMC, where a lower bound function can be obtained by fixing either term. Cluster and matching can be iteratively updated as follows:

- Infer the best cluster based on current matching results.

$$\begin{aligned} \mathbf{C}^{(t)} = &\arg\max_{\mathbf{C}} \sum_{ij} \mathbf{C}_{ij}\text{vec}(\mathbf{X}_{ij}^{(t-1)})^\top \mathbf{K}_{ij}\text{vec}(\mathbf{X}_{ij}^{(t-1)}) \\ s.t.\ \ &\mathbf{C}_{ik}\mathbf{C}_{kj} \le \mathbf{C}_{ij}, \forall i,j,k,\ \texttt{SCC}(\mathbf{C}) = N_c. \end{aligned} \tag{5}$$

- Maximize the affinity of graph pairs within clusters.

$$\begin{aligned} \mathbb{X}^{(t)} = &\arg\max_{\mathbb{X}} \sum_{ij} \mathbf{C}_{ij}^{(t)}\text{vec}(\mathbf{X}_{ij})^\top \mathbf{K}_{ij}\text{vec}(\mathbf{X}_{ij}) \\ s.t.\ \ &\mathbf{X}_{ij}\mathbf{1}_{n_j} \le \mathbf{1}_{n_i}, \mathbf{X}_{ij}^\top \mathbf{1}_{n_i} \le \mathbf{1}_{n_j}. \\ &\mathbf{X}_{ik}\mathbf{X}_{kj} \le \mathbf{X}_{ij}, \forall \mathbf{C}_{ik}^{(t)} = \mathbf{C}_{kj}^{(t)} = \mathbf{C}_{ij}^{(t)} = 1. \end{aligned} \tag{6}$$

Its convergence is easy to prove (shown in supplementary material). However, the performance of the algorithm is relatively poor in practice. It mainly attributes to its hard assignment restrictions on clustering. 1) Only the matchings in the same cluster are optimized while others stay unchanged. Consequently, the cluster inference tends to be the same, and the algorithm converges very fast to a sub-optimal. 2) The way to obtain cluster results is also not reliable, *e.g.* Spectral Clustering [23] is influenced by random initialization and obtains undesirable results especially in first few iterations when the matching is not good enough.

Therefore, we relax the constraints on cluster indicator $\mathbf{C}$. The original hard assignment constraint is:

$$\mathbf{C}_{ik}\mathbf{C}_{kj} \le \mathbf{C}_{ij}, \forall i,j,k,\ \texttt{SCC}(\mathbf{C}) = N_c. \tag{7}$$

And we provide two relaxed constraints which only focus the number of graph pairs in the indicator:

$$\begin{aligned} &\sum_{ij} \mathbf{C}_{ij} \le r\cdot N^2 \quad \text{(global constraint)}; \\ &\sum_{j} \mathbf{C}_{ij} \le r\cdot N, \forall i \quad \text{(local constraint)}. \end{aligned} \tag{8}$$

where $r \in [0, 1]$ is a hyper-parameter for ratio of chosen pairs. The global version limits the overall number of graph pairs and the local version sets the constraints for each graph. Relaxed constraints ignore the transitive relation of graph pairs $\mathbf{C}_{ik}\mathbf{C}_{kj} \leq \mathbf{C}_{ij}$ and judge each pair individually, which leads to a more flexible structure on indicator $\mathbf{C}$. We call the $\mathbf{C}$ obtained with relaxed constraints as **relaxed indicator**. The relaxed indicator enjoys two advantages:

1) Compared with the hard cluster or tree based structure in DPMC, the relaxed indicator allows for exploring more matching information, which may find better solution.

2) The relaxed indicator facilitates the pseudo label selection in unsupervised learning. It chooses the graph pair with higher affinity score, and thus prompts the quality of pseudo label. Details will be introduced in Sec. 4.3.

Note the indicator is relaxed during optimization, we further solve Eq. 6 with the converged matching results to obtain the final cluster division.

4.2. Majorization-minimization style M3C

We will show our proposed method: M3C in this subsection with three parts: initialization, cluster-step, and matching-step. The detailed algorithm is shown in Alg. 1.

1) Initialization. We obtain the initial matching $\mathbb{X}^0$ via RRWM [5] for its cost-effectiveness.

2) Clustering-Step. Two versions of relaxed indicator are proposed in Sec. 4.1, and the objective becomes:

$$\begin{aligned} \mathbf{C}^{(t)} &= \arg\max_{\mathbf{C}} \sum_{ij} \mathbf{C}_{ij} \mathrm{vec}(\mathbf{X}_{ij}^{(t-1)})^\top \mathbf{K}_{ij} \mathrm{vec}(\mathbf{X}_{ij}^{(t-1)}) \\ s.t. \quad & \sum_{ij} \mathbf{C}_{ij} \leq r \cdot N^2 \quad or \quad \sum_{j} \mathbf{C}_{ij} \leq r \cdot N, \forall i \end{aligned} \tag{9}$$

The solution of Eq. 9 is straightforward. Since $\mathbf{X}_{ij}^{(t-1)}$ are fixed, the affinity score $\mathrm{vec}(\mathbf{X}_{ij}^{(t-1)})^\top \mathbf{K}_{ij} \mathrm{vec}(\mathbf{X}_{ij}^{(t-1)})$ are also fixed. The key is to choose the graph pair with higher affinity score. As for global constraints, we rank all the graph pairs by affinity and set a threshold at highest rN^2. For local constraints, we rank the neighbors of each graph and take the top rN neighbors in the sense of k-nearest-neighbor. We call them **global-rank** and **local-rank**.

In practice, we also find a better ranking that combines the global constrains and local constraints to some degree, so-called **bi-rank**. More specifically, we specify the rank of an pair graph $(\mathcal{G}_u, \mathcal{G}_v) \in$ as: $R_{uv} = i + j$ where $\mathcal{G}_u$ is the i-th nearest neighbor of $\mathcal{G}_v$ and $\mathcal{G}_v$ is the j-th nearest neighbor of $\mathcal{G}_u$. Then we take a threshold on all the graph pairs based on $\{R_{uv}\}$. In this sense of bi-directional ranking, the indicator $\mathbf{C}$ reflects the affinity relationship among graphs and could satisfy the desirable neighbor constraints.

3) Matching-Step. The original goal of the matching-step is to maximize the pairwise affinity within clusters.

Algorithm 1: Learning-free solver M3C for joint matching and clustering of graphs

Input: Iterations number T, affinity matrix set $\mathbb{K}$, strategy $\{$*global-rank*, *local-rank*, *bi-rank*$\}$.

1 Obtain initialization matching $\mathbb{X}^{(0)}$ via RRWM [5];
2 **for** $i = 1 : T$ **do**
  /* Clustering-Step */
3 Based on $\mathbb{X}^{(i-1)}$, obtain $\mathbf{C}^i$ via solving Eq. 9 via global-rank, local-rank or bi-rank (see Sec. 4.2);
  /* Matching-Step */
4 Based on $\mathbf{A}^{(i)}$, obtain $\mathbb{X}^i$ via solving Eq. 10 to find optimal matching composition on $\mathbf{A}^{(i)}$;
5 **end**
6 Obtain cluster result $\mathbf{C}^t$ by spectral clustering [23] algorithm on $\mathbb{X}^t$ and $\mathbf{K}$;

Output: Matching $\mathbb{X}$, cluster $\mathbf{C}$.

However, as the indicator has been relaxed, the optimization format needs to be generalized too.

The essential optimization rule we need to follow is to optimize the graph pairs of the same category. For relaxed indicator, a pair of graphs are more likely to belong to the same category if there are many paths from one graph to another. Following such assumption, we adopt MGM-Floyd, an algorithm that aims to find optimal matching composition on supergraph, to solve the matching-step. The optimization target turns out to be:

$$\mathbb{X}^{(t)} = \arg\max_{\mathbb{X} \in \mathbf{P}_{\mathbf{C}^{(t)}}} \sum_{ij} \mathrm{vec}(\mathbf{X}_{ij})^\top \mathbf{K}_{ij} \mathrm{vec}(\mathbf{X}_{ij}) \tag{10}$$

where the relaxed indicator $\mathbf{C}^{(t)}$ is treated as the adjacency of supergraph, and $\mathbf{P}_{\mathbf{C}^{(t)}}$ denotes all the paths on it. As one can see, more paths between $\mathcal{G}_i$ and $\mathcal{G}_j$ will leads to a larger solution space $\mathbf{P}_{\mathbf{C}^{(t)}}(i, j)$ and thus the affinity is more likely to be optimized to a better stage.

Specifically, when $\mathbf{C}^{(t)}$ degenerates to the cluster division, the optimization target is the same as Eq. 6.

- For graph pairs of different categories, there is no path between them and they will not be optimized at all.
- For pairs of the same category, the optimal solution $\mathbf{X}_{ij}^{opt}$ of Eq. 6 will not contain the matching crossing the class. That is to say, it is also in $\mathbf{P}_{\mathbf{C}^{(t)}}(i, j)$, which leads to the same return.

Therefore, Eq. 10 is a well-defined generalization of Eq. 6. Meanwhile, all the pairs have a chance to be updated, which perturb the relaxed indicator $\mathbf{C}$ and avoid the algorithm falling into sub-optimal in certain cases.

4.3. Unsupervised Learning Model: UM3C

In this subsection, we will introduce our unsupervised learning model **UM3C**. We add the edge-wise affinity learn-

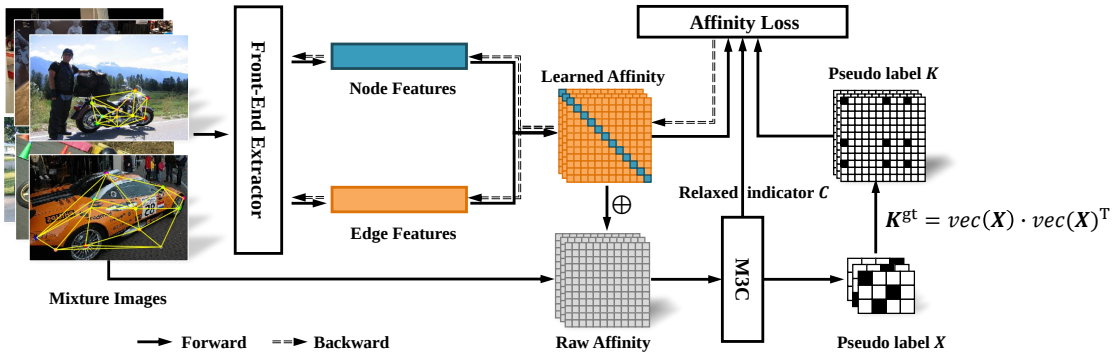

Figure 2. Pipeline for unsupervised learning model: **UM3C**. Mixture of images with different classes are input where key point coordinates are given and graphs are built with Delaunay triangulation [6]. Node features and edge features are extracted by VGG-16 [30] and Spline CNN [11]. The final affinity $\mathbf{K}$ is obtained by adding learned affinity $\mathbf{K}_{learn}$ and raw affinity $\mathbf{K}_{raw}$ (see Eq. 12). **M3C** solve the MGMC on $\mathbf{K}$ and N_c to gain the pseudo matching $\mathbf{X}^{gt}$, as well as the pseudo affinity $\mathbf{K}^{gt}$. Affinity loss selects the pseudo pair based on relaxed indicator $\mathbf{C}$ and encourages the learned affinity matrix to get close to $\mathbf{K}^{gt}$. The gradient back-propagation is shown by the arrows.

ing to the model and expand pipeline [33] from KAP [19] to QAP [21] formulation. Pseudo label selection based on relaxed indicator is introduced to facilitate unsupervised learning. The overall pipeline is shown in Fig. 2.

4.3.1 Edge-wise Affinity Learning

UM3C adopts the standard front-end extractor [27]: both node and edge features are extracted by VGG-16 backbone [30] and refined by Spline CNN [11]. Unlike previous work, we build the learned affinity matrix as:

$$\mathbf{K}_{learn} = \mathbf{F}_1^\top \mathbf{F}_2, \tag{11}$$

where $\mathbf{F}_1, \mathbf{F}_2$ denotes node or edge feature from $\mathcal{G}_1$ and $\mathcal{G}_2$. The operator Λ to adjust the scale between nodes is removed for training stability. On the other side, the raw affinity $\mathbf{K}_{raw}$ is constructed as a standard process, which encodes geometric information with Gaussian kernel. The final affinity consists of two parts where their weight is equal as in general we have no prior on their importance:

$$\mathbf{K} = \mathbf{K}_{learn} + \mathbf{K}_{raw}. \tag{12}$$

Normalization is applied to make elements in affinity between 0 and 1. In the first few epochs, raw affinity $\mathbf{K}_{raw}$ plays a leading role to prompt the quality of pseudo matching $\mathbf{X}^{gt}$. In the later stages of training, the quality of learned affinity $\mathbf{K}_{learn}$ will outperform the $\mathbf{K}_{raw}$, and the final affinity can be further improved.

Pseudo matching $\mathbf{X}^{gt}$ is obtained via **M3C**. However, $\mathbf{X}^{gt} \in \{0,1\}^{n_1 \times n_2}$ cannot serve as pseudo label for learned affinity $\mathbf{K}_{learn} \in \mathbb{R}^{n_1 n_2 \times n_1 n_2}$ directly. Therefore, we transfer the pseudo matching into pseudo affinity as:

$$\mathbf{K}^{gt} = \mathrm{vec}(\mathbf{X}^{gt}) \cdot \mathrm{vec}(\mathbf{X}^{gt})^\top. \tag{13}$$

The affinity loss is designed based on Element-wise Binary Cross Entropy Loss, as also adopted in [31, 33].

$$\mathtt{Loss}(\mathbf{K}, \mathbf{K}^{gt}) = \sum_{st} \mathbf{K}^{gt}_{pq} \log(\mathbf{K}_{pq}) + (1 - \mathbf{K}^{gt}_{pq}) \log(1 - \mathbf{K}_{pq}) \tag{14}$$

Note $\mathbf{K}_{pq}$ denotes the element of p-th row and q-th column rather than the whole affinity matrix between $\mathcal{G}_p$ and $\mathcal{G}_q$.

4.3.2 Pseudo Label Selection

Moreover, we also optimize the pseudo pair selection for loss based on the guidance of relaxed indicator $\mathbf{C}$.

Note the previous work GANN [33] adds up the loss of all the pairs within the inferred cluster. However, one wrong assignment in the cluster leads to many pairs of pseudo labels of different categories, which can degrade the quality of $\mathbf{X}^{gt}$. Meanwhile, matching accuracy within clusters can not be guaranteed even they are of the same category, either.

Instead of using all the pairs in the cluster, **UM3C** follows the relaxed indicator $\mathbf{C}$ and selects the graph pairs with higher affinity rank as pseudo labels. With the assumption that a higher affinity score denotes higher matching accuracy, the quality of our pseudo affinity $\mathbf{K}^{gt}$ can be largely improved. The overall loss is designed as:

$$\mathtt{Loss} = \sum_{ij} \mathbf{C}_{ij} \cdot \mathtt{BCELoss}(\mathbf{K}_{ij}, \mathbf{K}^{gt}_{ij}) \tag{15}$$

It selects $\mathbf{X}^{gt}_{ij}$ with higher accuracy, which makes pseudo affinity label more close to ground truth. Consequently, the stability and training convergence speed are also improved.

5. Experiments

5.1. Protocols

Experiments of traditional learning-free solvers are conducted on a laptop of 2.30GHz 4-core CPU and 16GB mem-

$N_c \times N_g$	3×8, 0 outlier					3×8, 2 outliers					3×8, 4 outliers				
Metrics	MA ↑	CA ↑	CP ↑	RI ↑	time(s) ↓	MA ↑	CA ↑	CP ↑	RI ↑	time(s) ↓	MA ↑	CA ↑	CP ↑	RI ↑	time(s) ↓
RRWM [5]	0.748	0.815	0.879	0.871	**0.4**	0.595	0.541	0.643	0.68	**0.4**	0.572	0.547	0.661	0.685	**0.6**
CAO-C [37]	0.875	0.860	0.908	0.903	3.3	**0.727**	0.574	0.678	0.704	3.7	**0.661**	0.562	0.674	0.695	4.9
MGM-Floyd [17]	0.879	**0.931**	**0.958**	**0.952**	2.0	0.716	0.564	0.667	0.696	2.3	0.653	0.580	0.690	0.708	2.9
DPMC [35]	0.872	0.890	0.931	0.923	1.2	0.672	0.617	0.724	0.733	1.4	0.630	0.600	0.707	0.722	2.3
M3C (ours)	**0.884**	0.911	0.941	0.938	0.5	0.687	**0.653**	**0.750**	**0.758**	0.6	0.635	**0.646**	**0.748**	**0.753**	1.0
GANN [33]	0.896	0.963	0.976	0.970	5.2	0.593	0.871	0.903	0.888	31.3	0.189	0.674	0.708	0.709	108.9
UM3C (ours)	**0.955**	**0.983**	**0.988**	**0.988**	**3.2**	**0.866**	**0.944**	**0.964**	**0.956**	**3.5**	**0.848**	**0.96**	**0.977**	**0.967**	**4.2**

Table 1. Evaluation of matching and clustering metric with inference time for the mixture graph matching and clustering. Learning-free methods are compared on the top, while unsupervised models are shown on the bottom.

ory and implemented by Matlab R2020a. Experiments on learning methods are conducted on Linux workstation with Xeon-3175X@3.10GHz, RX8000, and 128GB memory.

Datasets. Willow ObjectClass [4] contains images from Caltech-256 [13] and Pascal VOC 2007 [10], which consists of 256 images from 5 categories: 40 cars, 40 motorbikes, 50 ducks, 66 winebottles, and 109 faces. Each image is labeled with 10 keypoints and we randomly add outliers for different experiment settings. The graph is built by sparse Delaunay triangulation. For learning-based models, we follow the standard process and crop the images to the bounding boxes of the objects and rescale to 256×256 pixel. For mixture mode, we sample N_c categories and randomly pick N_g graphs for each category and mix them as input.

We follow the standard protocol [17, 35, 37, 38] to construct affinity matrix $\mathbf{K}$, which re=weights the length affinity and angle affinity $\mathbf{K} = \beta\mathbf{K}_{\text{length}} + (1-\beta)\mathbf{K}_{\text{angle}}$. Note that this affinity is used in learning-free experiments as well as the $\mathbf{K}_{raw}$ term in unsupervised learning process.

Peer Methods. We run experiments on both learning-free and learning-based settings to evaluate our method M3C and UM3C. Note that we apply the **bi-rank** scheme for both M3C and UM3C if not otherwise specified.

For learning-free experiments, we mainly compare our methods M3C to two competitive multi-graph matching solvers *e.g.* DPMC [35] and MGM-Floyd [17] where DPMC is the SOTA solver of MGMC, and MGM-Floyd is the SOTA solver of multi-graph matching.

For learning-based experiments, we compare UM3C with unsupervised GANN [33] as well as two SOTA supervised models: BBGM [27] and NGMv2 [32].

Evaluation Metrics. Denote a cluster with a set of graphs $\mathcal{C} = \{\mathcal{G}_1 \dots \mathcal{G}_n\}$. Performance metrics include both matching accuracy and clustering quality:

- **Matching Accuracy (MA)** [35] We only consider the intra-cluster matching accuracy and thus by adapting the accuracy for single cluster, we have $\text{MA} = \frac{1}{\sum \mathbf{C}_{ij}} \sum_{ij} \mathbf{C}_{ij} \cdot \texttt{ACC}(\mathbf{X}_{ij})$, where $\texttt{ACC}(\mathbf{X}_{ij})$ denotes accuracy for matching $\mathbf{X}_{ij}$. Here $\mathbf{C}$ refer to a indicator for strict cluster division.

- **Clustering Purity (CP)** [22]: it is given by $\text{CP} = \frac{1}{N}\sum_{i=1}^{N_c} \max_{j\in\{1,\dots,N_c\}} |\mathcal{C}'_i \cap \mathcal{C}_j|$ where $\mathcal{C}'_i$ is the predicted cluster i and $\mathcal{C}_j$ is the ground truth cluster j, and N is the total number of graphs.

- **Rand Index (RI)** [26]: $\text{RI} = \frac{n_{11}+n_{00}}{n} \in [0,1]$ where, n_{11} represents the number of graphs predicted in the same cluster with same label, n_{00} the number of pairs that are in different clusters with different labels, and normalized by the total number of graph pairs n.

- **Clustering Accuracy (CA)** [35], it is defined by:

$$\text{CA} = 1 - \frac{1}{N_c}\left(\sum_{\mathcal{C}_a}\sum_{\mathcal{C}'_a \neq \mathcal{C}'_b} \frac{|\mathcal{C}'_a \cap \mathcal{C}_a|\,|\mathcal{C}'_b \cap \mathcal{C}_a|}{|\mathcal{C}_a|\,|\mathcal{C}_a|} + \sum_{\mathcal{C}'_a}\sum_{\mathcal{C}_a \neq \mathcal{C}_b} \frac{|\mathcal{C}'_a \cap \mathcal{C}_a|\,|\mathcal{C}'_a \cap \mathcal{C}_b|}{|\mathcal{C}_a|\,|\mathcal{C}_b|}\right)$$

  where $\mathcal{C}_a, \mathcal{C}_b$ are ground truth clusters and $\mathcal{C}'_a, \mathcal{C}'_b$ denotes the predicted cluster.

5.2. Performance on MGMC

We first compare our M3C and UM3C with peer MGM methods GAGM [33], DPMC [35], MGM-Floyd [17], CAO [37], two-graph matching baseline RRWM [5], and unsupervised learning method GANN [33], following the setting given in [33]. We tested these methods with $N_c = 3, N_g = 8$ where car, motorbike, and duck are used. Outliers are randomly generated on the graphs and we exclude outliers when reporting matching accuracy.

Table 1 shows that our learning-free solver M3C reaches competitive results on each metric with a relatively low time cost compared to all the learning-free algorithms. It has the best matching accuracy (over 1% gain) and reaches a high clustering accuracy (only $1-2\%$ loss compared to the best method) on the setting without outlier. On the setting with 2 and 4 outliers, it has $3-5\%$ gain on clustering accuracy and $3-4\%$ loss on matching accuracy. Overall,M3C has the most balanced performance, showing its robustness against outliers on both matching and clustering problems.

Meanwhile, the unsupervised model UM3C significantly outperforms all the methods listed: it achieves 6% to 18%

$N_c \times N_g$	3×20, 2 outliers		4×20, 2 outliers		5×20, 2 outliers		5×15, 2 outliers		5×10, 2 outliers		$3 \times [20, 10, 5]$, 2 outliers	
Metrics	MA ↑	CA ↑	MA ↑	CA ↑	MA ↑	CA ↑	MA ↑	CA ↑	MA ↑	CA ↑	MA ↑	CA ↑
RRWM [5]	0.658	0.932	0.642	0.858	0.665	0.790	0.648	0.693	0.664	0.679	0.633	0.681
CAO-C [37]	0.849	0.946	0.812	0.855	0.820	0.790	0.801	0.708	0.804	0.679	0.787	0.757
MGM-Floyd [17]	0.845	0.945	0.812	0.878	0.819	0.807	0.798	0.727	0.799	0.707	0.778	0.755
DPMC [35]	**0.867**	0.942	0.827	0.894	0.775	0.772	0.739	0.713	0.756	0.744	**0.795**	0.823
M3C (ours)	0.857	**0.961**	**0.851**	**0.933**	**0.835**	**0.900**	**0.812**	**0.805**	**0.809**	**0.780**	0.792	**0.881**
GANN [33]	0.465	0.810	0.570	0.793	0.508	0.765	0.522	0.784	0.506	0.807	0.430	0.757
UM3C (ours)	**0.888**	**0.983**	**0.891**	**0.960**	**0.875**	**0.971**	**0.874**	**0.951**	**0.870**	**0.966**	**0.857**	**0.965**

Table 2. Evaluation of matching and clustering accuracy by varying the number of clusters, and number of graphs in each cluster. Learning-free methods are compared on the top, while unsupervised models are shown at the bottom.

Metrics	MA ↑	CA ↑	CP ↑	RI ↑
vanilla GANN [33]	0.896	0.963	0.976	0.970
change to M3C (ours)	0.910	0.955	0.969	0.967
+ Spline CNN	0.912	0.954	0.969	0.967
+ Label selection	0.922	0.964	0.978	0.971
+ Edge-wise affinity	0.955	0.983	0.988	0.988

Table 3. Ablation study of M3C by adding components. We train and test each model under $N_c = 3, N_g = 8$ without outlier.

improvement in matching accuracy together with 5% to 30% boost in clustering accuracy on these settings.

5.3. Varying Cluster Number and Cluster Size

Here we test the generalization ability against the number of graphs. For $N_c = 3$, the categories including car, motorbike, and winebottle are used. While for $N_c = 4$, car, motorbike, winebottle, and face are chosen. Unbalanced cluster size is also tested with $N_c = 3$ which involves 20 cars, 15 motorbikes, and 5 wine bottles. GANN and UM3C are both trained under $N_c = 5, N_g = 10$ without outlier. During the testing phase, we randomly add 2 outliers to the graph across all the settings, if not otherwise specified.

Table 2 verifies the robustness of our methods with state-of-the-arts on different cluster and graph numbers. Our learning-free solver M3C maintains a competitive performance compared to DPMC, with 1% loss to 2% gain in matching accuracy and 2% to 9% increase in clustering accuracy. Such achievements also reflect the superiority of our proposed ranking scheme.

UM3C outperforms all peer methods in all metrics. Specifically, it maintains a cluster accuracy over 0.95 and a matching accuracy over 0.85 on all settings which is a $2 - 7\%$ improvement in matching accuracy and $2 - 18\%$ growth in clustering accuracy. Since the training setting of UM3C is different from the testing, it justifies good generalization ability of our method over various graph numbers, clusters, and outliers. Moreover, even our method is trained under a simpler setting, it can also be deployed into a more complicated scenario and achieve satisfying performance.

5.4. Ablation Study

We conduct ablation study of UM3C on $N_c = 3, N_g = 8$ without outlier to examine the effectiveness of each part of our model. The baseline method is constructed by replacing the GAGM solver in GANN with M3C. Since GANN doesn't use Spline CNN to refine its feature, we hereby test how the introduction of Spline CNN would influence the unsupervised learning method. The effectiveness of label selection and edge-wise affinity learning is validated by adding each component successively. Table 3 shows that the learning of edge-wise affinity contributes significantly to our model by an improvement of 3%. This shows the shortage of raw affinity and the necessity of learning edge-wise affinity. Both Spline CNN and label selection help to improve our learning process. Note that our baseline method has already surpassed the performance of GANN. This also verifies the effectiveness of our solver **M3C**.

5.5. Comparison with Supervised Models

We compare with two peer supervised learning methods, BBGM [27] and NGMv2 [32]. Both methods are learned under the two-graph matching setting, and we train different models for different numbers of outliers. The clustering metric is based on their learned affinity score, and the clustering algorithm follows the same protocol as our methods. Results are shown in Fig. 3. Our unsupervised learning method even outperforms the supervised model by a notable margin, which suggests the effectiveness of feature extraction by **UM3C**. In particular, we draw two insights: 1) MGMC is a multi-graph matching problem. Trained on the pairwise setting, supervised models lack a mechanism to utilize information from other graphs, and thus leads to a gap in matching accuracy. 2) M3C is a cluster-aware solver which enlarges the affinity gap between pairs of the same or different classes. BBGM and NGM aim to maximize all the pairwise affinity for all the graph pairs, which degrades the quality of clustering metrics.

5.6. Comparison of Different Ranking Schemes

We compare three proposed ranking schemes under different test settings. M3C-Global, M3C-Local, and M3C-Bi

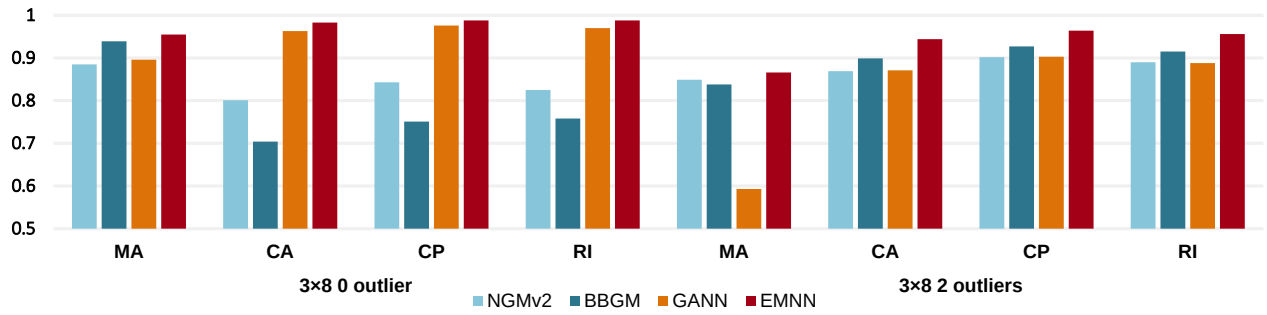

Figure 3. Comparison with peer supervised learning methods, under 3 clusters with 8 graphs for each cluster. Outlier number is 0 and 2.

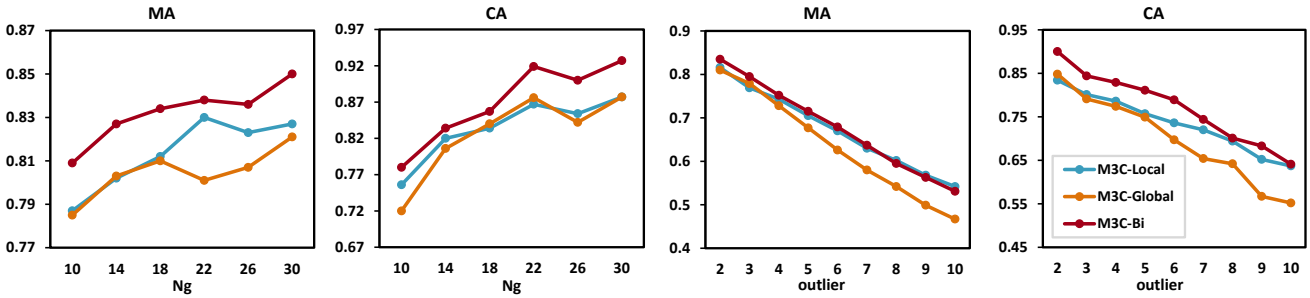

Figure 4. Comparison of three ranking schemes (local, global, bi) by varying number of graphs in each cluster and number of outliers.

refer to global-rank, local-rank, and bi-rank, respectively. We first test them by varying the graph number N_g in each cluster where the cluster number $N_c = 5$ stays the same. We also exam these methods by adding different numbers of outliers to the graph where $N_c = 5, N_g = 20$. From Fig. 4, we find M3C-Bi achieves the best performance among these methods. Thus we choose M3C-Bi as the solver of our unsupervised learning model UM3C. These tests also verify that both global and local ranking schemes are good approximations of the optimization target. Moreover, it confirms our ranking methods have strong robustness and generalization ability. Even there exist 10 outliers, we still achieve a matching accuracy over 0.5 and clustering accuracy over 0.65. Meanwhile, as the number of graphs increases, our method's performance also increases by 4% in matching accuracy and 15% in clustering accuracy. This also explains why M3C doesn't exceed other learning-free solvers in Table 1 but significantly outperforms them in Table 2.

5.7. Time Complexity Analysis

The speedup over DPMC [35] and vanilla MGM-Floyd [17] benefits from a lower time complexity bound. Let N and n denote the number of graphs and nodes, respectively. It costs $\mathcal{O}(N^2n^3)$ to calculate the score matrix and $\mathcal{O}(N^2 \log N^2)$ to construct the supergraph. In the worst case, we would add $\frac{N(N-1)}{2}$ edges to the supergraph, which leads to $\mathcal{O}(N^3n^3)$ time cost for performing MGM-Floyd. However, the expectation of the number of edges added is much smaller than the worst case. In fact, it is smaller than $\frac{N_cN_g(N_g-1)}{2}$ in most cases where N_c denotes the number of clusters and N_g denotes the maximum number of graphs in these clusters. Meanwhile, no need to calculate the consistency when performing our algorithms also lowers our time cost. These all make the actual running time significantly less than DPMC and vanilla MGM-Floyd. Such improvements also lead to less training and inference time.

6. Conclusion and Outlook

This paper has presented a principled approach to jointly solve the graph matching and clustering tasks for graphs from a mixture of modes, which is a realistic problem beyond classic graph matching. Our learning-free solver M3C fits the majorization-minimization nature by introducing a relaxed cluster indicator. Meanwhile, we also combine our M3C solver with unsupervised learning pipeline and propose the unsupervised version UM3C. It even outperforms the supervised models on joint graph matching and clustering, which is a practical and challenging task.

However, there are still several **limitations** of our work and can be further improved for future work. 1) Although relaxed, the cluster indicator is still discrete while the ranking scheme is a bit simple. It can be further relaxed to continuous space, where the optimization objective is weighted so the solver can integrate more the pairwise matching results during optimization. 2) For unsupervised learning pipeline, the hand-craft affinity affects the quality of pseudo labels, which can degrade notably for challenging datasets *e.g.* PascalVOC [9]. It hurts the stability of the training process, and thus the model fails to converge. More robust and reliable methods are in need for the unsupervised pipeline.

References

[1] Yunsheng Bai, Hao Ding, Song Bian, Ting Chen, Yizhou Sun, and Wei Wang. Simgnn: A neural network approach to fast graph similarity computation. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, pages 384–392, 2019. 2

[2] Alexander C Berg, Tamara L Berg, and Jitendra Malik. Shape matching and object recognition using low distortion correspondences. In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, volume 1, pages 26–33. IEEE, 2005. 1

[3] Yuxin Chen, Leonidas Guibas, and Qixing Huang. Near-optimal joint object matching via convex relaxation. In Eric P. Xing and Tony Jebara, editors, *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 100–108, Bejing, China, 22–24 Jun 2014. PMLR. 2

[4] Minsu Cho, Karteek Alahari, and Jean Ponce. Learning graphs to match. In *ICCV*, 2013. 6

[5] Minsu Cho, Jungmin Lee, and Kyoung Mu Lee. Reweighted random walks for graph matching. In *European conference on Computer vision*, pages 492–505. Springer, 2010. 2, 4, 6, 7

[6] Boris Delaunay et al. Sur la sphere vide. *Izv. Akad. Nauk SSSR, Otdelenie Matematicheskii i Estestvennyka Nauk*, 7(793-800):1–2, 1934. 1, 5

[7] M Fatih Demirci, Ali Shokoufandeh, Yakov Keselman, Lars Bretzner, and Sven Dickinson. Object recognition as many-to-many feature matching. *International Journal of Computer Vision*, 69(2):203–222, 2006. 1

[8] Kun Duan, Devi Parikh, David Crandall, and Kristen Grauman. Discovering localized attributes for fine-grained recognition. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3474–3481. IEEE, 2012. 1

[9] Mark Everingham, Luc Gool, Christopher K. Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *IJCV*, 88(2):303–338, June 2010. 8

[10] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes challenge 2007 (voc2007) results. 2007. 6

[11] Matthias Fey, Jan Eric Lenssen, Frank Weichert, and Heinrich Müller. Splinecnn: Fast geometric deep learning with continuous b-spline kernels. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 869–877, 2018. 5

[12] Michael Goesele, Noah Snavely, Brian Curless, Hugues Hoppe, and Steven M Seitz. Multi-view stereo for community photo collections. In *2007 IEEE 11th International Conference on Computer Vision*, pages 1–8. IEEE, 2007. 1

[13] Gregory Griffin, Alex Holub, and Pietro Perona. Caltech-256 object category dataset. Technical report, California Institute of Technology, 2007. 6

[14] N. Hu, Q. Huang, B. Thibert, and L. Guibas. Distributable consistent multi-graph matching. *CVPR*, 2018. 2

[15] Qi-Xing Huang and Leonidas Guibas. Consistent shape maps via semidefinite programming. In *Computer Graphics Forum*, volume 32, pages 177–186. Wiley Online Library, 2013. 2

[16] David R. Hunter and Kenneth L. Lange. A tutorial on mm algorithms. *The American Statistician*, 58:30 – 37, 2004. 3

[17] Zetian Jiang, Tianzhe Wang, and Junchi Yan. Unifying offline and online multi-graph matching via finding shortest paths on supergraph. *TPAMI*, 43(10):3648–3663, 2021. 1, 2, 3, 6, 7, 8

[18] V. G. Kim, W. Li, N. J. Mitra, S. DiVerdi, and T. Funkhouser. Exploring collections of 3d models using fuzzy correspondences. In *SIGGRAPH*, 2012. 2

[19] T. C. Koopmans and M. Beckmann. Assignment problems and the location of economic activities. *Econometrica*, pages 53–76, 1957. 5

[20] Spyridon Leonardos, Xiaowei Zhou, and Kostas Daniilidis. Distributed consistent data association via permutation synchronization. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2645–2652. IEEE, 2017. 2

[21] Eliane Maria Loiola, Nair Maria Maia de Abreu, Paulo Oswaldo Boaventura-Netto, Peter Hahn, and Tania Querido. A survey for the quadratic assignment problem. *EJOR*, pages 657–90, 2007. 5

[22] Christopher D Manning, Hinrich Schütze, and Prabhakar Raghavan. *Introduction to information retrieval*, volume 39. Cambridge University Press, 2008. 6

[23] Andrew Y Ng, Michael I Jordan, and Yair Weiss. On spectral clustering: Analysis and an algorithm. In *Advances in neural information processing systems*, pages 849–856, 2002. 3, 4

[24] Deepti Pachauri, Risi Kondor, and Vikas Singh. Solving the multi-way matching problem by permutation synchronization. In *Advances in neural information processing systems*, pages 1860–1868. Citeseer, 2013. 2

[25] James Petterson, Jin Yu, Julian J McAuley, and Tibério S Caetano. Exponential family graph matching and ranking. In *Advances in Neural Information Processing Systems*, pages 1455–1463, 2009. 1

[26] William M Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical association*, 66(336):846–850, 1971. 6

[27] Michal Rolínek, Paul Swoboda, Dominik Zietlow, Anselm Paulus, Vít Musil, and Georg Martius. Deep graph matching via blackbox differentiation of combinatorial solvers. In *European Conference on Computer Vision*, pages 407–424. Springer, 2020. 2, 5, 6, 7

[28] Dinggang Shen and Christos Davatzikos. Hammer: hierarchical attribute matching mechanism for elastic registration. *IEEE transactions on medical imaging*, 21(11):1421–1439, 2002. 1

[29] Ian Simon, Noah Snavely, and Steven M Seitz. Scene summarization for online image collections. In *2007 IEEE 11th International Conference on Computer Vision*, pages 1–8. IEEE, 2007. 1

[30] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2014. 5

[31] Runzhong Wang, Junchi Yan, and Xiaokang Yang. Learning combinatorial embedding networks for deep graph matching. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3056–3065, 2019. 2, 5

[32] R. Wang, J. Yan, and X. Yang. Combinatorial learning of robust deep graph matching: an embedding based approach. *IEEE TPAMI*, 2020. 2, 6, 7

[33] Runzhong Wang, Junchi Yan, and Xiaokang Yang. Graduated assignment for joint multi-graph matching and clustering with application to unsupervised graph matching network learning. In *NeurIPS*, 2020. 1, 2, 5, 6, 7

[34] Runzhong Wang, Junchi Yan, and Xiaokang Yang. Neural graph matching network: Learning lawler's quadratic assignment problem with extension to hypergraph and multiple-graph matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021. 2

[35] Tianzhe Wang, Zetian Jiang, and Junchi Yan. Clustering-aware multiple graph matching via decayed pairwise matching composition. *AAAI*, 2020. 1, 2, 6, 7, 8

[36] Tao Wang, He Liu, Yidong Li, Yi Jin, Xiaohui Hou, and Haibin Ling. Learning combinatorial solver for graph matching. In *CVPR*, pages 7568–7577, 2020. 2

[37] Junchi Yan, Minsu Cho, Hongyuan Zha, Xiaokang Yang, and Stephen M Chu. Multi-graph matching via affinity optimization with graduated consistency regularization. *IEEE transactions on pattern analysis and machine intelligence*, 38(6):1228–1242, 2015. 1, 2, 3, 6, 7

[38] J. Yan, J. Wang, H. Zha, X. Yang, and S. Chu. Consistency-driven alternating optimization for multigraph matching: A unified approach. *TIP*, 24(3):994–1009, 2015. 1, 2, 3, 6

[39] Tianshu Yu, Runzhong Wang, Junchi Yan, and Baoxin Li. Deep latent graph matching. In *International Conference on Machine Learning*, pages 12187–12197. PMLR, 2021. 2

[40] Andrei Zanfir and Cristian Sminchisescu. Deep learning of graph matching. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2684–2693, 2018. 2

[41] Xiaowei Zhou, Menglong Zhu, and Kostas Daniilidis. Multi-image matching via fast alternating minimization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4032–4040, 2015. 2