Unifying Offline and Online Multi-graph Matching via Finding Shortest Paths on Supergraph

Zetian Jiang, Tianzhe Wang, and Junchi Yan* *Member, IEEE,*

Abstract—This paper addresses the problem of multiple graph matching (MGM) by considering both offline batch mode and online setting. We explore the concept of cycle-consistency over pairwise matchings and formulate the problem as finding optimal composition path on the supergraph, whose vertices refer to graphs and edge weights denote score function regarding consistency and affinity. By our theoretical study we show that the offline and online MGM on supergraph can be converted to finding all pairwise shortest paths and single-source shortest paths respectively. We adopt the Floyd algorithm [1] and shortest path faster algorithm (SPFA) [2], [3] to effectively find the optimal path. Extensive experimental results show our methods surpass state-of-the-art MGM methods, including CAO [4],MISM [5], IMGM [6] and many other recent methods in offline and online settings. Source code will be made publicly available.

Index Terms—Graph Matching, Multiple Graph Matching, Online Graph Matching, Shortest Path Search

1 Introduction

Graph matching (GM) refers to finding node correspondences among two or multiple graphs given an affinity model. Different from point matching [7] that usually only considers the node-wise unary affinity, GM models the second-order [8] or higher-order information [9] for more robust matching against local noise. Due to its NP-complete nature, existing methods mostly seek approximate solutions [8], [10], [11], [12]. For two-graph matching, a general form can be written as the following quadratic assignment programming (QAP) problem [13] which is also called Lawler's QAP [14] and will be the focus of this paper. Other forms e.g. Koopmans-Beckmann's QAP [13] can be regarded as a special case for Lawler's QAP form.

$$J(\mathbf{X}) = \min_{\mathbf{X}\in\{0,1\}^{n_1\times n_2}} \mathrm{vec}(\mathbf{X})^\top \mathbf{K}\mathrm{vec}(\mathbf{X}) \tag{1}$$

where $\mathbf{X}$ is a (partial) permutation matrix indicating the node correspondence, and $\mathbf{K} \in \mathbb{R}^{n_1n_2\times n_1n_2}$ is the affinity matrix whose diagonal (off-diagonal) encodes the node-to-node affinity (edge-to-edge affinity) between two graphs. The symbol $\mathrm{vec}(\cdot)$ here denotes the column-wise vectorization of the input matrix.

Though there are extensive works on two-graph matching [8], [15], [16], the trend is (arguably) shifted to multiple graph matching (MGM). The reasons may include: i) information can be fused as more graphs are used; ii) practical problems often involve more than two graphs, either in an offline batch mode [4] or its online incremental setting. In particular, the problem for incremental multiple graph matching (IMGM) (a.k.a. online MGM) has recently received attention [6] for its practical utility in that graphs are often collected sequentially from time to time. For instance, in visual SLAM or odometry, the images are often captured over time for sequential matching [17]. The previously found protein may be added with newly discovered one after a long time. Similar case also holds for graphics dataset collection which may take years of continuous efforts. Given the existing graphs, IMGM tries to efficiently and effectively match the new coming graphs by exploring the previous matching results.

This paper considers the problem settings for both offline joint matching and online incremental matching. Specifically, the offline mode refers to the traditional multiple graph matching scenario whereby all the graphs are processed in a batch for joint matching. For the later case i.e. online setting, it means the graphs are sequentially arriving and incremental matching is needed to handle the newly arrivals. Most previous works are devoted to the offline case [4], [18], [19], [20], [21], [22], [23], [24], [25] while there is little study [6] on the latter one. We take a supergraph view on the initial matchings between two graphs where each vertex in the supergraph denotes the graph for matching and the edges can carry the pairwise matching information. The composition chain over pairwise matchings can be regarded as a path in the supergraph. Under this setting, we show for the first time, the offline MGM can be formulated as a shortest path problem and solved by dynamic programming i.e. Floyd algorithm [1]. Moreover, we find also for the first time, the online MGM can be formulated as the single-source shortest path problem and it can be solved by shortest path faster algorithm (SPFA) [2], [3]. In contrast, the most related work [4] ignores the dynamic programming nature of the composition chain based mechanism for offline MGM, which leads to a naive and inefficient composition path finding method, so for its incremental version [6] which reuses the offline MGM solver [4] in an out-of-box fashion.

Contributions. The main contributions of this paper are:

1) By taking a supergraph view on MGM whereby pairwise matching composition can be regarded as path on supergraph, we formulate MGM into a shortest path finding problem. We draw careful numerical analysis on the behavior of the composition strategy, and then devise a pairwise matching updating method based on the classic Floyd algorithm with dynamic programming, termed **MGM-Floyd**. It outperforms CAO [4] notably.

2) For online MGM, we devise a new method inspired by the classic shortest path faster algorithm (SPFA) [26]. It can effectively utilize the existing matchings to match new coming graphs. This method is termed by **MGM-SPFA**, which is shown more accurate than the state-of-the-art incremental matching method

Z. Jiang and T. Wang are with Zhiyuan College, Shanghai Jiao Tong University, Shanghai, 200240, China. E-mail: {maple_jzt, usedtobe}@sjtu.edu.cn,
J. Yan (correspondence author) is with Department of Computer Science and Engineering, and MoE Key Lab of Artificial Intelligence, AI Institute, Shanghai Jiao Tong University, 200240, China. E-mail: yanjunchi@sjtu.edu.cn.
Z. Jiang and T. Wang contribute equally to this work.

IMGM [6], as the only available baseline for comparison.

3) Based on MGM-SPFA, we propose a fast version by clustering graphs and performing MGM-SPFA in each cluster. The resulting method is called **FastSPFA** being more efficient than MGM-SPFA and more accurate than IMGM.

2 Related Work

We focus on literature on MGM. Readers are referred to [27] for a more comprehensive review for graph matching.

For multi-graph matching, researchers often turn into regularization models and one widely adopted and studied regularizer in literature is *cycle consistency* [4], [6], [28]. In general it refers to the fact that the bijection correspondence between graph G_i and G_j shall be consistent with a derived one through an intermediate graph G_k: namely $\mathbf{X}_{ij} = \mathbf{X}_{ik}\mathbf{X}_{kj}$. Obviously consistency is the necessary condition and becomes an important cue for perfect bijections among multiple graphs, especially given the affinity in practice is often biased due to noise and other reasons.

There are different ways of utilizing the cycle consistency regularizer, and here we review three representative lines of works. Methods in the first group directly use the affinity of each pair of graphs together with the consistency regularizer (or hard consistency constraint) for optimization. In [28], the consistency is strictly obeyed by using a set of basis $\{\mathbf{X}_{rk}\}_{k=1}^{N}$ encoding two-graph matchings such that other matching can be represented by the cycle: $\mathbf{X}_{ij} = \mathbf{X}_{ri}^{\top}\mathbf{X}_{rj}$. In this sense, the reference graph G_r can be regarded as a central base. In contrast to this centralized framework, a more distributed approach is devised in [4]. In this work, treating all two-graph matchings $\{\mathbf{X}_{ij}\}_{i,j=1,1}^{N,N}$ as the solution to compute, the affinity score and consistency are jointly modeled in the objective function for optimization. As such, consistency plays a soft regularizer.

While the second group involves methods e.g. [29], [30], [31] try to take the initial (noisy) pairwise matching result as input, and try to recover cycle-consistent solutions by post-processing. Spectral techniques [19], [20], [32] are developed to extract the consistent matches by the spectrum (top eigenvectors) of the matrix composed of all putative matches. The underlying rationale is that the problem can be formulated as quadratic integer programming which can be relaxed into a generalized Rayleigh problem [20]. In the seminal work [19], the authors show theoretical conditions for exact recovery. They present a convex relaxation method for estimating cycle-consistent matchings by finding the nearest positive semidefinite matrix to the input matrix stacking by all initial matchings. Improvement is done in [21] by assuming the underlying rank of the variable matrix can be estimated reliably, allowing for partial matching which is further improved by MatchALS [22]. One drawback of [21], [22] is the use of an excessively large n^+ to accommodate a flexible universe incurring high computational cost. One natural idea is to extract the $\overline{n} < n^+$ common inliers. In this regard, [5] solves a new factorization model whereby a scalable block coordinate descent technique is developed. There are also methods addressing the distributed multi-graph matching problem. [25] presents a greedy construction method for graph clustering with theoretical support. In [24], the authors devise a decentralized version of spectral method [20]. In a more recent work [33], message-passing algorithm is adopted to solve the MGM problem which accounts for first-order/second-order affinity as well as cycle consistency for optimization.

For the setting of incremental multi-graph matching (IMGM) for graphs arriving sequentially, it receives little attention until a very recent approach as presented in [6]. Incremental data association is also considered in [34], however only first-order node-wise similarity is considered, and the proposed method is only applied on small-scale data.

The most related methods to ours are the CAO-C (and its derivatives CAO-UC, CAO-PC) [4] and the incremental method IMGM [6]. They both adopt the composition based strategy to generate new pairwise matchings which is the basic step also used in our approach. While our methods significantly differ from them as we present a more principled perspective based on graph theory (Floyd and SPFA) to effectively find the composition path. Our methods are grounded by our careful theoretical analysis which is new in literature. Based on these findings, we develop methods for both offline and online MGM under a unified perspective.

3 Proposed Unified Approaches

We first quote and rewrite the consistency definitions in [4], [6].

Definition 1. **[4]** *Given N graphs $\{G_k\}_{k=1}^{N}$ and pairwise matchings $\mathbb{X} = \{\mathbf{X}_{ij}\}_{i=1,j=i+1}^{N-1,N}$, define G_k's unary consistency:*

$$C_u(k,\mathbb{X}) = 1 - \frac{\sum_{i=1}^{N-1}\sum_{j=i+1}^{N}\|\mathbf{X}_{ij} - \mathbf{X}_{ik}\mathbf{X}_{kj}\|_F/2}{nN(N-1)/2} \in (0,1]. \tag{2}$$

Definition 2. **[4]** *Given $\{G_k\}_{k=1}^{N}$ and matching configuration $\mathbb{X}$, for any pair G_i and G_j, the pairwise consistency is defined as:*

$$C_p(\mathbf{X}_{ij},\mathbb{X}) = 1 - \frac{\sum_{k=1}^{N}\|\mathbf{X}_{ij} - \mathbf{X}_{ik}\mathbf{X}_{kj}\|_F/2}{nN} \in (0,1]. \tag{3}$$

Definition 3. **[4]** *Given N graphs $\{G_k\}_{k=1}^{N}$ and $\mathbb{X}$, define the overall consistency as:*

$$C(\mathbb{X}) = \frac{\sum_{k=1}^{N} C_u(k,\mathbb{X})}{N} \in (0,1]. \tag{4}$$

3.1 Motivation and theoretical study

Similar to [4], we take a supergraph view on MGM, which is grounded on our theoretical study. One shall note the contribution of this paper has nothing to do with the formulation of supergraph, rather it lies in our theoretical analysis and the resulting efficient and effective algorithms for MGM.

In a supergraph $\mathcal{H} = \{V = \{G_1,\dots,G_N\}, E = \{\mathbb{X}\}\}$, the vertices and edges denote graphs for matching G, and pairwise matchings $\mathbf{X}$. One aims to seek the optimal $\mathbf{X}_{ik_1}\mathbf{X}_{k_1k_2}\cdots\mathbf{X}_{k_nj}$ composition to generate (improved) $\mathbf{X}_{ij}$. Since $\mathcal{H}$ is a complete graph, each pairwise matching composition can be viewed as a path from G_i to G_j via $G_{k_1}, G_{k_2}\dots, G_{k_n}$. Finding the optimal composition of $\mathbf{X}_{ij}$ on $\mathbb{X}$ is equivalent to searching the optimal path from G_i to G_j on $\mathcal{H}$.

For finding an optimal path, it is necessary to define the quality of a matching. We use the score function in [4] for its generality:

$$S(\mathbf{X}_{ij},\mathbb{X}) = \overbrace{(1-\lambda)J(\mathbf{X}_{ij})}^{\text{affinity score}} + \overbrace{\lambda C_p(\mathbf{X}_{ij},\mathbb{X})}^{\text{pairwise consistency}}, \tag{5}$$

which consists of both affinity score and pairwise consistency (see Definition 3). In the rest of the paper, for brevity we omit $\mathbb{X}$ in S without ambiguity. To some extent, $S(\mathbf{X}_{ij})$ can be regarded as the distance or weight of edge on $\mathcal{H}$.

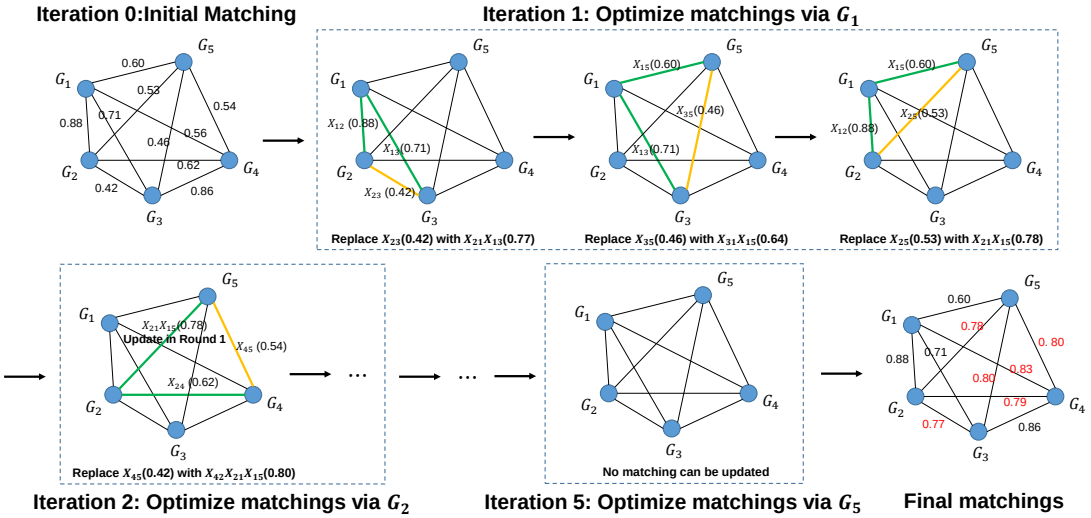


Fig. 1. Illustration of MGM-Floyd for offline MGM. The green and yellow line denotes the matchings used to update and need to be updated, respectively. The number in the brackets on each edge denotes the matching score and the composition matching is near the number with X_{ij} being the initial matching. In iteration i, use G_i as internal vertex to update the matchings.

Assumption 1. *Given proper λ and relatively good initial $\mathbb{X}$, suppose S obey the following property:*

$$S(\mathbf{X}_{ij}) < S(\mathbf{X}'_{ij}) \Longrightarrow S(\mathbf{X}_{ij}\mathbf{X}_{jk}) < S(\mathbf{X}'_{ij}\mathbf{X}_{jk}),$$

In other words, if $S(\mathbf{X}_{ij}) < S(\mathbf{X}'_{ij})$, replacing $\mathbf{X}_{ij}$ in any composition path with $\mathbf{X}'_{ij}$ can lead to better path: we have no worry about optimizing the path between G_i and G_j will affect the optimal solution between the other two vertices.

As one can see, Assump. 1 is not that straightforward. Therefore, we give two propositions to support it. Since S consists of both affinity and consistency, we prove the similar property of these two components in Prop. 1 and Prop. 2.

Proposition 1. *Let μ be a uniform distribution of all possible permutation matrices. If $J(\mathbf{X}_{ij}) > J(\mathbf{X}'_{ij})$, it holds that $\mathbb{E}_{X_{jk}\sim\mu}[J(\mathbf{X}_{ij}\mathbf{X}_{jk})] > \mathbb{E}_{X_{jk}\sim\mu}[J(\mathbf{X}'_{ij}\mathbf{X}_{jk})]$.*

Proposition 2. *If $C_p(\mathbf{X}_{ij}, \mathbb{X}) < C_p(\mathbf{X}'_{ij}, \mathbb{X})$, we claim that $C_p(\mathbf{X}_{ij}\mathbf{X}_{jk}, \mathbb{X}) < C_p(\mathbf{X}'_{ij}\mathbf{X}_{jk}, \mathbb{X})$.*

If Prop. 1 and Prop. 2 both hold, it is instinctive that Assump. 1 will also hold in most cases. The core idea for our proposed method is that: based on this rather mild assumption, we can approximately treat the -$S(\mathbf{X}_{ij})$ as the distance between G_i and G_j and convert the optimal path finding problem into solving the shortest path problem. We will introduce the Floyd and SPFA to solve the offline and online MGM in Sec. 3.2 and 3.3, respectively.

Here we give the proof of Prop. 1 and Prop. 2. We first present a moderate assumption on which classical graph matching problem stands, to facilitate our numerical analysis:

Assumption 2. *The affinity is a monotonic increasing function w.r.t. accuracy (F-norm distance between ground truth, defined in Sec. 4.1) i.e.: $J(\mathbf{X}_{ij}) < J(\mathbf{X}'_{ij})$ means that*

$$\left\|\mathbf{X}_{ij} - \bar{\mathbf{X}}_{ij}\right\|_F > \left\|\mathbf{X}'_{ij} - \bar{\mathbf{X}}_{ij}\right\|_F$$

where $\bar{\mathbf{X}}$ denotes the ground truth permutation matrix and $\|A\|_F = \sqrt{\Sigma A_{ij}^2}$ is the Frobenius norm.

Now, we can prove the two propositions as follows.

Proof of proposition 1. First of all, we try to simplify the conclusion that we need to prove. For convenience, we leave out the $\mathbf{X}_{jk} \sim \mu$ constraint for all expectations. According to Assump. 2, to prove $\mathbb{E}[J(\mathbf{X}_{ij}\mathbf{X}_{jk})] > \mathbb{E}[J(\mathbf{X}'_{ij}\mathbf{X}_{jk})]$, we only need to show that $\mathbb{E}[\|\mathbf{X}_{ij}\mathbf{X}_{jk} - \bar{\mathbf{X}}_{ik}\|_F] < \mathbb{E}[\|\mathbf{X}'_{ij}\mathbf{X}_{jk} - \bar{\mathbf{X}}_{ik}\|_F]$.

Denote $M(\mathbf{X}_{ij})$ as the number of matched nodes in $\mathbf{X}_{ij}$, we can derive that $\|\mathbf{X}_{ij} - \bar{\mathbf{X}}_{ij}\|_F = \sqrt{2(n - M(\mathbf{X}_{ij}))}$. Obviously, the formula turns into $\mathbb{E}[M(\mathbf{X}_{ij}\mathbf{X}_{jk})] > \mathbb{E}[M(\mathbf{X}'_{ij}\mathbf{X}_{jk})]$ and we prove it as following:

Without loss of generality, suppose each graph has n nodes, and the ground truth matching is always $\bar{\mathbf{X}}_{ij} = I$. Denote $p = \frac{M(\mathbf{X}_{ij})}{n}, q = \frac{M(\mathbf{X}'_{ij})}{n}, s = \frac{M(\mathbf{X}_{jk})}{n}$. By Assump. 2 and $J(\mathbf{X}_{ij}) > J(\mathbf{X}'_{ij})$, we have:

$$\|\mathbf{X}_{ij} - I\|_F < \|\mathbf{X}'_{ij} - I\|_F. \tag{6}$$

This leads to $\sqrt{2(1-p)n} < \sqrt{2(1-q)n}$ or equivalently $p > q$.

We can first calculate the value of $\mathbb{E}[M(\mathbf{X}_{ij}\mathbf{X}_{jk})]$. Denote Z_t as the indicator of matching situation of t-th node in G_i, if it matches t-th node in G_k, $Z_t = 1$. Otherwise, $Z_t = 0$. We have:

$$\begin{aligned}
\mathbb{E}[M(\mathbf{X}_{ij}\mathbf{X}_{jk})] =& \sum_t \mathbb{E}[Z_t] \\
=& \sum_t \big(Pr[\mathbf{X}_{ij}(t,t) = 1 \cap \mathbf{X}_{jk}(t,t) = 1] \\
&+ \sum_{t' \neq t} Pr[\mathbf{X}_{ij}(t,t') = 1 \cap \mathbf{X}_{jk}(t',t) = 1]\big) \\
=& n\left(ps + \frac{(1-p)(1-s)}{n-1}\right) \\
=& \frac{n}{n-1}\left(psn + p + s + 1\right).
\end{aligned}$$

Likewise, the expectation of matching nodes number in $\mathbf{X}_{ij}\mathbf{X}_{jk}$ is $\mathbb{E}[M(\mathbf{X}'_{ij}\mathbf{X}_{jk})] = \frac{n}{n-1}(qsn + q + s + 1)$. Thus:

$$\mathbb{E}[M(\mathbf{X}_{ij}\mathbf{X}_{jk})] - \mathbb{E}[M(\mathbf{X}'_{ij}\mathbf{X}_{jk})] = \frac{sn-1}{n-1}(p-q)n.$$

In the real case, the initial matching accuracy can easily reach over 40%, which means $s \geq \frac{1}{n}$ always holds. Therefore, we have

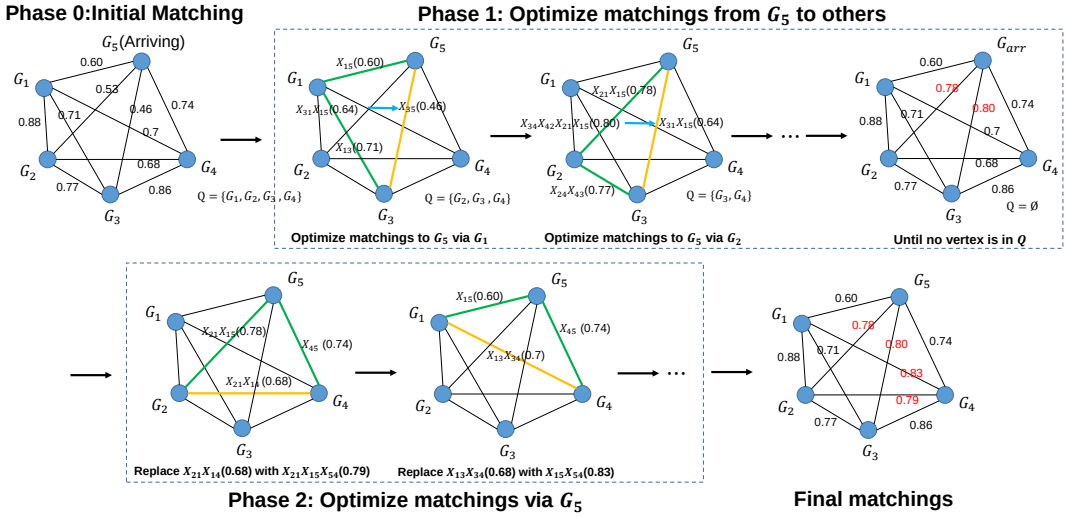


Fig. 2. Illustration of MGM-SPFA for online MGM. Here G_5 is the arriving graph. The green and yellow line denotes the matchings used to update and need to be updated, respectively. The number in the brackets on each edge denotes the matching score and the composition matching is near the number with X_{ij} being the initial matching. In Phase 1, fix all the matchings in $\mathcal{H}\backslash\{G_5\}$ and update each pair of $\mathbf{X}_{iN}$ using other vertices. In Phase 2, fix all X_{iN} and use G_N as internal vertex to update all the matchings $\mathbf{X}_{ij}$ in $\mathcal{H}\backslash\{G_5\}$.

$\frac{sn-1}{n-1}(p-q)n > 0$ and thus $\mathbb{E}[J(\mathbf{X}_{ij}\mathbf{X}_{jk})] > \mathbb{E}[J(\mathbf{X}'_{ij}\mathbf{X}_{jk})]$. □

Moreover, one can see that the boost of accuracy expectation depends on the following two factors:

1) The matching performance gap of $\mathbf{X}_{ij}$ and $\mathbf{X}'_{ij}$, i.e. $p-q$.
2) The performance of transition matching $\mathbf{X}_{jk}$, i.e. s.

As affinity is a monotonic increasing function w.r.t accuracy (according to Assump. 2), the boost of affinity expectation would largely depend on these two factors, too. This indicates that when optimizing a certain matching, higher overall accuracy also facilitates raising the performance of composition matching methods. On the contrary, if the overall accuracy is poor, the chance of the boost on performance could be relatively slim. That is why we asked for 'a relatively good initial matching' in Assump. 1

Proof of proposition 2. By the definition of $C_p(\mathbf{X}_{ij}, \mathbb{X})$, we have:

$$\sum_{t=1}^{N} \|\mathbf{X}_{ij} - \mathbf{X}_{it}\mathbf{X}_{tj}\|_F > \sum_{t=1}^{N} \|\mathbf{X}'_{ij} - \mathbf{X}_{it}\mathbf{X}_{tj}\|_F .$$

As $\mathbf{X}_{jk}$ is permutation matrix (invariant to the F-norm). So,

$$\sum_{t=1}^{N} \|(\mathbf{X}_{ij} - \mathbf{X}_{it}\mathbf{X}_{tj})\mathbf{X}_{jk}\|_F > \sum_{t=1}^{N} \|(\mathbf{X}'_{ij} - \mathbf{X}_{it}\mathbf{X}_{tj})\mathbf{X}_{jk}\|_F$$
$$C_p(\mathbf{X}_{ij}\mathbf{X}_{jk}, \mathbb{X}) < C_p(\mathbf{X}'_{ij}\mathbf{X}_{jk}, \mathbb{X}).$$

□

3.2 Offline multiple graph matching

In view of supergraph, offline MGM aims to seek an optimal composition path of initial matchings between any two vertices on the supergraph. CAO [4] is such a searching process that iteratively finds the best intermediates to improve the composition path. However, the path length found by this method would not exceed 2^T, where T is the number of iterations. To overcome this limitation, we propose a dynamic programming based method MGM-Floyd based on Floyd shortest path algorithm to solve the offline MGM. Compared with [4], MGM-Floyd is able to find the optimal composition path more efficiently with fewer comparisons and thus being more competitive.

Algorithm 1: MGM-Floyd (Offline MGM)

Input: affinity matrix $\{\mathbf{K}_{ij}\}_{i,j=1}^{N}$, initial $\mathbb{X}^{(0)}$, λ.
1 Set consistency weight to 0 for affinity based boosting.
2 **for** *each graph* G_v **do**
3 **for** *each pair of graphs* G_x, G_y **do**
4 set $S_{org} = S(\mathbf{X}_{xy})$ by Eq. 5 (or using approximate $S^{\mathbb{X}}_{pc}$ by Eq. 7, $S^{\mathbb{X}}_{uc}$ by Eq. 8 for speedup);
5 set $S_{opt} = S(\mathbf{X}_{xv}\mathbf{X}_{vy})$ by Eq. 5 (or Eq. 7, Eq. 8);
6 **if** $S_{org} < S_{opt}$ **then**
7 $\mathbf{X}_{xy} \leftarrow \mathbf{X}_{xv}\mathbf{X}_{vy}$;
8 Set consistency weight to $\lambda > 0$ and run Line 2-7 again;
Output: optimized matching $\mathbb{X}$.

Let $\mathbf{X}_{ij}^{(k)}$ denote the optimal composition path from G_i to G_j only using the vertices in the set $\{G_1, G_2 \dots G_k\}$ as intermediate graphs along the way. The initial configuration $\mathbf{X}_{ij}^{(0)}$ is obtained by two-graph matching solver e.g. [8]. $\mathbf{X}_{ij}^{(k)}$ can only be transferred from the two states:

1) From $\mathbf{X}_{ij}^{(k-1)}$, the optimal path contains no G_k,
2) From $\mathbf{X}_{ik}^{(k-1)}\mathbf{X}_{kj}^{(k-1)}$, the optimal path contains G_k.

Thus, we can derive the transition equation by comparing the S-value of two possible states:

$$\mathbf{X}_{ij}^{(k)} = \arg\max_{\mathbf{X}\in\Theta} (1-\lambda)J(\mathbf{X}) + \lambda C_p(\mathbf{X}, \mathbb{X}),$$

where $\Theta = \{\mathbf{X}_{ik}^{(k-1)}\mathbf{X}_{kj}^{(k-1)}, \mathbf{X}_{ij}^{(k-1)}\}$. Besides, $\mathbb{X}^{(k)}$ only depends on $\mathbb{X}^{(k-1)}$, so the transfer function can be optimized to effectively save memory cost:

$$\mathbf{X}_{ij} = \arg\max_{\mathbf{X}\in\{\mathbf{X}_{ik}\mathbf{X}_{kj}, \mathbf{X}_{ij}\}} (1-\lambda)J(\mathbf{X}) + \lambda C_p(\mathbf{X}, \mathbb{X}).$$

An illustrative working example is given in Fig. 1. The detailed process is described in Alg. 1.

According to Assump. 1 and Prop. 2, one can see that our methods work well when the initial matching is relatively high.

Algorithm 2: MGM-SPFA (Online MGM)

Input: Affinity matrix $\{\mathbf{K}_{ij}\}_{i,j=1}^{N}$, λ, initial matching $\mathbb{X}$.
1 **for** *each newly arriving graph* G_N **do**
2 use two-graph solver to obtain $\mathbb{X}_{Ni}^{(0)}$ for G_N and others;
3 initialize the graph queue $\mathcal{Q} = \{G_1, G_2 \cdots, G_{N-1}\}$;
4 **while** $\mathcal{Q}$ *is not empty* **do**
5 obtain G_x in $\mathcal{Q}$ and remove it;
6 **for** *each graph* G_y **do**
7 set $S_{org} = S(\mathbf{X}_{yN})$ by Eq. 5 (or Eq. 7, Eq. 8);
8 set $S_{opt} = S(\mathbf{X}_{yx}\mathbf{X}_{xN})$ by Eq. 5 (or Eq. 7, Eq. 8);
9 **if** $S_{org} < S_{opt}$ **then**
10 $\mathbf{X}_{yN} \leftarrow \mathbf{X}_{yx}\mathbf{X}_{xN}$;
11 add G_y into $\mathcal{Q}$;
12 **for** *each pair of graphs* G_x, G_y *in* $\mathcal{H}\backslash\{G_N\}$ **do**
13 set $S_{org} = S(\mathbf{X}_{xN})$ by Eq. 5 (or Eq. 7, Eq. 8);
14 set $S_{opt} = S(\mathbf{X}_{xy}\mathbf{X}_{yN})$ by Eq. 5 (or Eq. 7, Eq. 8);
15 **if** $S_{org} < S_{opt}$ **then**
16 $\mathbf{X}_{xy} \leftarrow \mathbf{X}_{xN}\mathbf{X}_{Ny}$;

Output: optimized matching $\mathbb{X}$.

Algorithm 3: FastSPFA (Online Fast MGM)

Input: Affinity matrix $\{\mathbf{K}_{ij}\}_{i,j=1}^{N}$, λ, C_{min}, initial $\mathbb{X}$.
1 **for** *each newly arriving graph* G_N **do**
2 calculate the cluster number $M = \max(1, N/C_{min})$.
3 partition $\mathcal{H}\backslash\{G_N\}$ into M clusters randomly.
4 **for** *each cluster* C_i **do**
5 generate the sub-supergraph $\mathcal{H}_i$ by $C_i \cup \{G_N\}$.
6 apply Line 1 to 16 in Alg. 2 to obtain the edges between G_N and other vertices on $\mathcal{H}_i$: $\{\mathbb{X}_{Nh}\}_{h=1}^{N-1}$.
7 perform the same post-processing Line 12 to 16 in Alg. 2 to optimize all pairwise matchings $\mathbf{X}_{uv}$ in $\mathcal{H}\backslash\{G_N\}$: i.e. use $\mathbf{X}_{uN}\mathbf{X}_{Nv}$ to update $\mathbf{X}_{uv}$ via G_N for each pair.

Output: optimized matching $\mathbb{X}$.

Therefore, we utilize affinity boost technique to improve the initial matching accuracy before we run our MGM-Floyd method. Specifically, we run the process above twice. In the first phase (affinity boosting phase), λ is set to 0 to obtain a better initial matching by only taking affinity into consideration. As consistency term is ignored, the accuracy is not high. Then we conduct the process again with a proper λ to take both affinity and consistency into account to obtain final matching (MGM-Floyd phase).

3.3 Online multiple graph matching

Online MGM aims at matching the arriving graph G_N to $N-1$ previous graphs which have already been matched. On supergraph $\mathcal{H}$, it is equivalent to find the optimal composition path from the arriving graph to the existing ones. Therefore, we propose an online method MGM-SPFA based on SPFA, a single-source shortest path algorithm.

The idea of MGM-SPFA is that when $\mathbf{X}_{Ni}$, the path from G_N to G_i is updated, G_i can be the intermediate point to improve the composition path of its adjacent vertices. We maintain a queue of all those vertices like G_i and check the path to their neighbors until no path can be updated. MGM-SPFA consists of three parts:

1) Add edge between arriving graphs G_N with the other graphs on supergraph $\mathcal{H}$, i.e. obtaining the initial matching $\mathbb{X}_{Ni}$ from G_N to the others by two-graph matching solver e.g. [35]. Add all vertices into queue as all paths to these vertices are updated.
2) Obtain the first vertex in the queue and update the composition path of its adjacent vertices. If any better path yields, add the endpoint into queue. Repeat till there is no vertex in the queue.
3) Optimize all the matchings by updating $\mathbf{X}_{uv}$ using $\mathbf{X}_{uN}\mathbf{X}_{Nv}$ for each pair of G_u and G_v in $\mathcal{H}\backslash\{G_N\}$.

Fig. 2 shows the whole process. Details can be found in Alg. 2.

3.4 Acceleration techniques

Speedup techniques are used for MGM-Floyd and MGM-SPFA.

Speedup by consistency approximation. This part follows the seminal work [4] as briefly described here. The most time-consuming part refers to calculating C_p in Eq. 5. Following the definitions in [4] as rewritten in Definition 1 and 2, we employ the efficient $S_{pc}^{\mathbb{X}}(\mathbf{X}_{ij}, \mathbf{X}_{jk})$, $S_{uc}^{\mathbb{X}}(\mathbf{X}_{ij}, \mathbf{X}_{jk})$ to approximate S [4]:

$$S_{pc}^{\mathbb{X}} = (1-\lambda)J(\mathbf{X}_{ij}\mathbf{X}_{jk}) + \lambda\sqrt{C_p(\mathbf{X}_{ij}, \mathbb{X})C_p(\mathbf{X}_{jk}, \mathbb{X})}, \quad (7)$$

$$S_{uc}^{\mathbb{X}} = (1-\lambda)J(\mathbf{X}_{ij}\mathbf{X}_{jk}) + \lambda C_u(j, \mathbb{X}). \quad (8)$$

Speedup by asynchronous update. Under such an approximation setting, one can then update $S_{pc}^{\mathbb{X}}$, $S_{uc}^{\mathbb{X}}$ asynchronously to reduce time cost. Specifically, when very few of the $\binom{N}{2}$ edges on $\mathcal{H}$ are updated, slight changes in C_p and C_u can be ignored. Therefore, we can refresh the consistency matrix at intervals instead of update lively to acceleration.

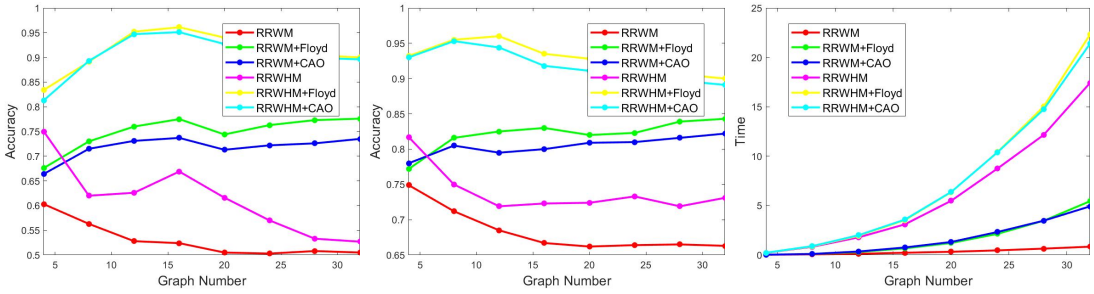
Speedup by clustering. As graphs accumulate, the time cost of MGM-SPFA increases dramatically. One natural idea is to divide the whole set into multiple clusters.

As the peer work [6] shows speedup by graph clustering, we hereby propose another online method FastSPFA, which boosts speed with a little accuracy loss, compared with MGM-SPFA. Specifically, FastSPFA consists of the following steps:

1) Similar to the initialization process of MGM-SPFA, we first add the edge i.e. initial matchings $\mathbb{X}_{Ni}$, between the arriving graph G_N with the others on $\mathcal{H}$ by two-graph matching solver e.g. [8], as discussed in Sec. 3.2.
2) Choose minimum allowed number C_{min} for a cluster and split the graphs into M clusters, such that $|\text{Cluster}_i| \geq C_{min}$ and $M = max(1, \lfloor \frac{N}{C_{min}} \rfloor)$. $\lfloor x \rfloor$ means rounding down x.
3) Run MGM-SPFA for each cluster C_i independently, to compute each graph G_u's matching against G_N: $\mathbf{X}_{Nu}$.
4) Perform the same post-processing in Sec. 3.3: use $\mathbf{X}_{uN}\mathbf{X}_{Nv}$ to update $\mathbf{X}_{uv}$ for each pair G_u and G_v in $\mathcal{H}\backslash\{G_N\}$.

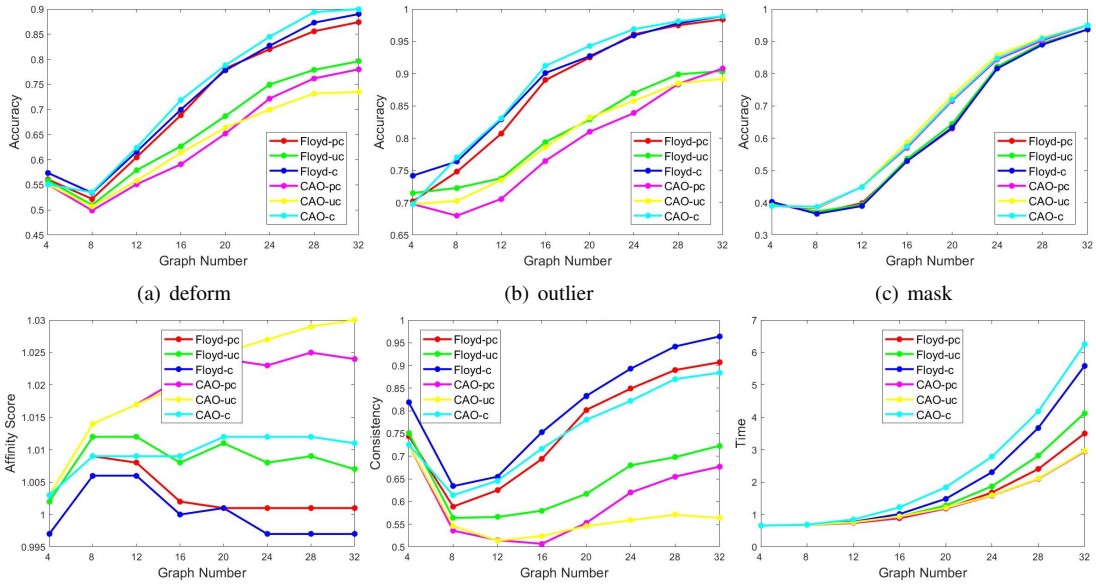
3.5 Disccusion with peer MGM methods

Finally we discuss the difference of recent MGM methods including those based on relaxation in continuous space. First we regard our methods as a discrete MGM method. This category also includes the composition based method CAO [4] and the tree structure based approach MatchOpt [28] that both iteratively transform the MGM problem into a two-graph matching one, and employ an off-the-shelf two-graph matching solver in each iteration to update the matchings, whereby the cycle consistency constraint is either automatically satisfied or encouraged. Another line of works [5], [19], [21] etc. try to solve the MGM problem in two stages. In the first stage, also a two-graph matching solver is adopted to obtain the initial (and possibly very inconsistent) pairwise matchings. Then in the second stage, the global consistency



(a) Car (b) Motorbike (c) Car

Fig. 3. Accuracy and time (in seconds) comparison of RRWM [8] and RRWHM [36] as the two-graph matching solver on Willow Object dataset.



(a) deform (b) outlier (c) mask

(d) deform (e) deform (f) deform

Fig. 4. Offline matching performance comparison on synthetic data (deform, outlier, complete) with different versions of MGM-Floyd and CAO [4].

is pursued via certain continuous optimization methods such as spectral clustering [20], semidefinite programming [19] and further robust improvement [21] using alternating direction methods of multipliers (ADMM) that can address the partial multi-graph matching setting. Differing from the above methods that separate the affinity based two-graph matching and consistency driven smoothing in two stages, some other works e.g. MatchALS [22] also explore the joint optimization regarding with consistency and affinity by relaxation in continuous space. The third thread of studies which are loosely related to ours is the rank-1/clustering based methods, whereby continuous based techniques like matrix decomposition [37], k-means clustering [38], and density-based clustering [39] are adopted. Note in these methods, often only node features are used rather than the edge information.

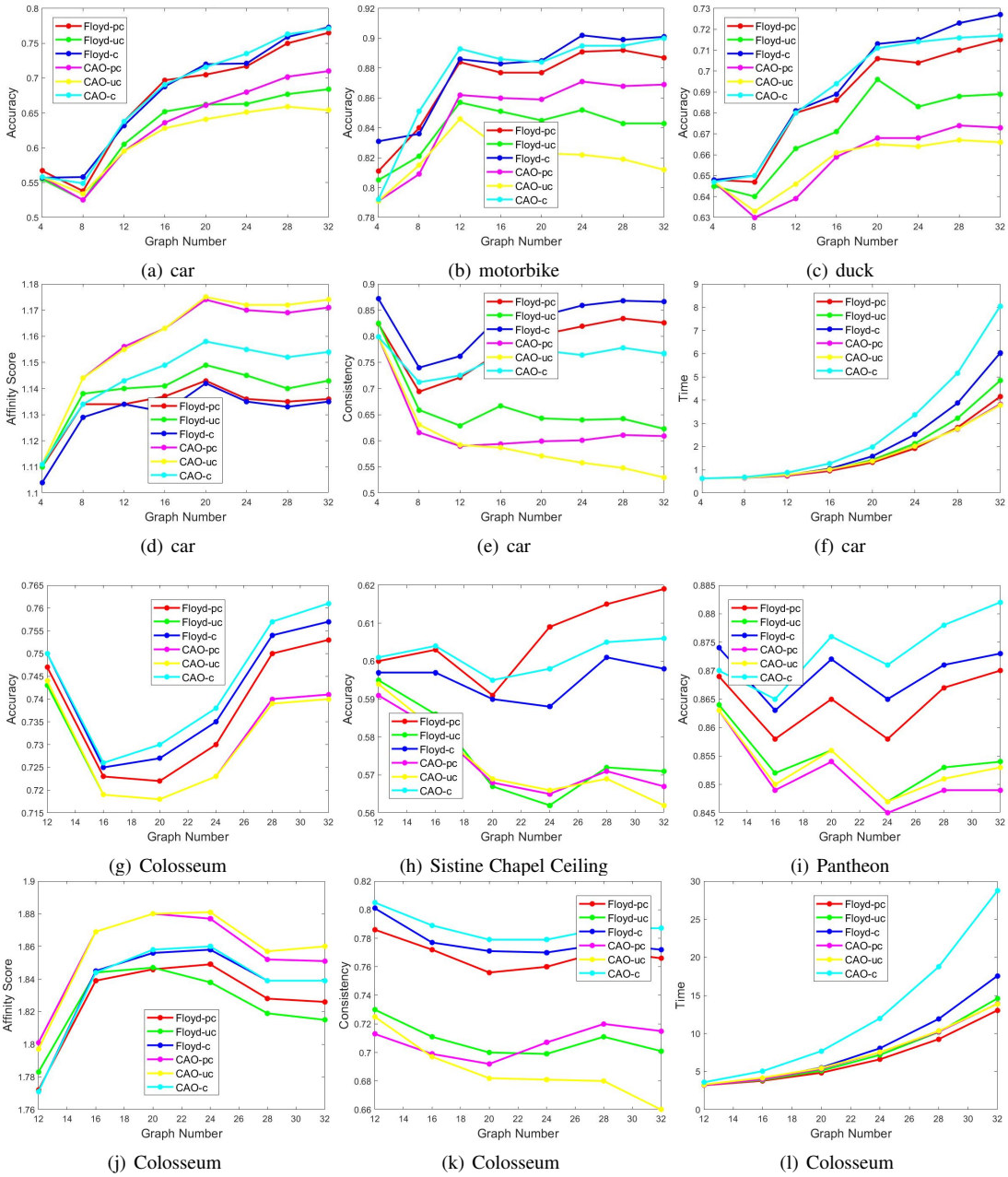
In our analysis, there are some limitations for continuous MGM methods: 1) the relaxation is an approximation to the raw matching problem which causes unwanted (even unbounded) model deviation, as discussed in two-graph matching [40]. This difficulty is pronounced for the complex MGM problem; 2) the relaxed optimization problem for MGM is still very challenging for its non-convexity, hard constraints, and high-dimensionality, which either cause scalability issue (e.g. MatchLift [21]), or limited exploration to geometric edge information (e.g. MatchALS [22]); 3) it still requires a post-processing step to round the continuous solution into (partial) permutation matrices, since the relaxation mostly cannot ensure a convergence to a matching solution.

In contrast, the discrete methods first do not suffer from the rounding issue. More specifically, we give a dynamic programming based algorithm to more effectively and efficiently explore the search space directly in the discrete space. The consistency constraints are basically automatically satisfied, rather than as a burden in the optimization objective which is often hard to optimize. Moreover, our discrete MGM approach can reused off-the-shelf two-graph matching solvers in an out-of-box manner regardless it is a discrete solver or a continuous one. In the following experiment part, we will show the efficacy of our approach in comparison with both continuous methods as well as previous discrete algorithms.

4 Experiments

4.1 Protocols

Experiments are performed on a desktop with 3.40 GHz 4-core CPU and 16G memory. Three popular metrics [4], [6] are used: accuracy, affinity score and consistency (abbreviated as acc, scr and con). $\text{acc} = 1 - \sum_{i,j} \|\mathbf{X}^*_{ij} - \mathbf{X}^{\text{GT}}_{ij}\|_F^2 / nN^2 \in [0, 1]$ refers to matching accuracy by comparing $\mathbf{X}^*_{ij}$ to ground-truth $\mathbf{X}^{\text{GT}}_{ij}$. The overall affinity score is $\text{scr} = \frac{1}{N^2} \sum_{i,j} \frac{\text{vec}(\mathbf{X}^*_{ij})^\top \mathbf{K}_{ij} \text{vec}(\mathbf{X}^*_{ij})}{\text{vec}(\mathbf{X}^{\text{GT}}_{ij})^\top \mathbf{K}_{ij} \text{vec}(\mathbf{X}^{\text{GT}}_{ij})}$. Consistency con refers to C_p in Definition 3. It is possible that $\text{scr} > 1$, as affinity function can be biased rendering an incorrect matching leads to a higher affinity score than the one of true match.



(a) car (b) motorbike (c) duck

(d) car (e) car (f) car

(g) Colosseum (h) Sistine Chapel Ceiling (i) Pantheon

(j) Colosseum (k) Colosseum (l) Colosseum

Fig. 5. Offline matching performance on objects from Willow-ObjectClass and Sub-Rome16K. Different versions of CAO and MGM-Floyd are tested.

For each comparison, 20 trials are tested by random sampling either from the synthetic data or real images, and the average of metrics are reported in the paper. Note that the time cost as shown in the plots regarding offline test is for the whole MGM, while that for online test is the incremental cost for the coming new graph. For all experiments, if not otherwise specified, we set $\lambda = 0.3$ and $\lambda = 0.4$ for MGM-Floyd and MGM-SPFA (FastSPFA) respectively. The test settings are depicted in Table 1.

Almost all the MGM methods need a two-graph matching solver as building block e.g. to obtain the initial $\mathbb{X}$. Here we mainly consider two choices: re-weighted random walk matching RRWM [8] and its hypergraph matching version RRWHM [36] which can more effectively explore the higher-order information in graph. We first give a comparison between these two solvers whose results are shown in Fig. 3. Without surprise, RRWHM based MGM can achieve better accuracy at the cost of more time overhead. Without loss of generality, in the rest of our experiments, we use RRWM as our basic solver for cost-effectiveness.

4.2 Datasets

Synthetic dataset. We first randomly generate a 'reference' adjacency matrix $\mathbf{E}^r$ of a complete graph. For each graph, the adjacency matrix is derived by further adding Gaussian perturbation and mask to each 'reference' edge. Specifically, the 'perturbed' edge weight q_{ij}^p is calculated by: $q_{ij}^p = q_{ij}^r + N(0, \epsilon)$, where ϵ is a hyperparameter of deformation. Moreover, a mask is constructed and imposed on the initial matching $\mathbb{X}$ controlled by random sampling $\rho \in [0, 1]$ via random sampling which means the ratio of all the $N \times N$ initial pairwise matchings that are generated by random permutation matrix. As ρ grows, the initial accuracy for $\mathbb{X}$ can decrease which we will use for studying the behavior of our methods when Assump. 1 does not hold. The edge affinity

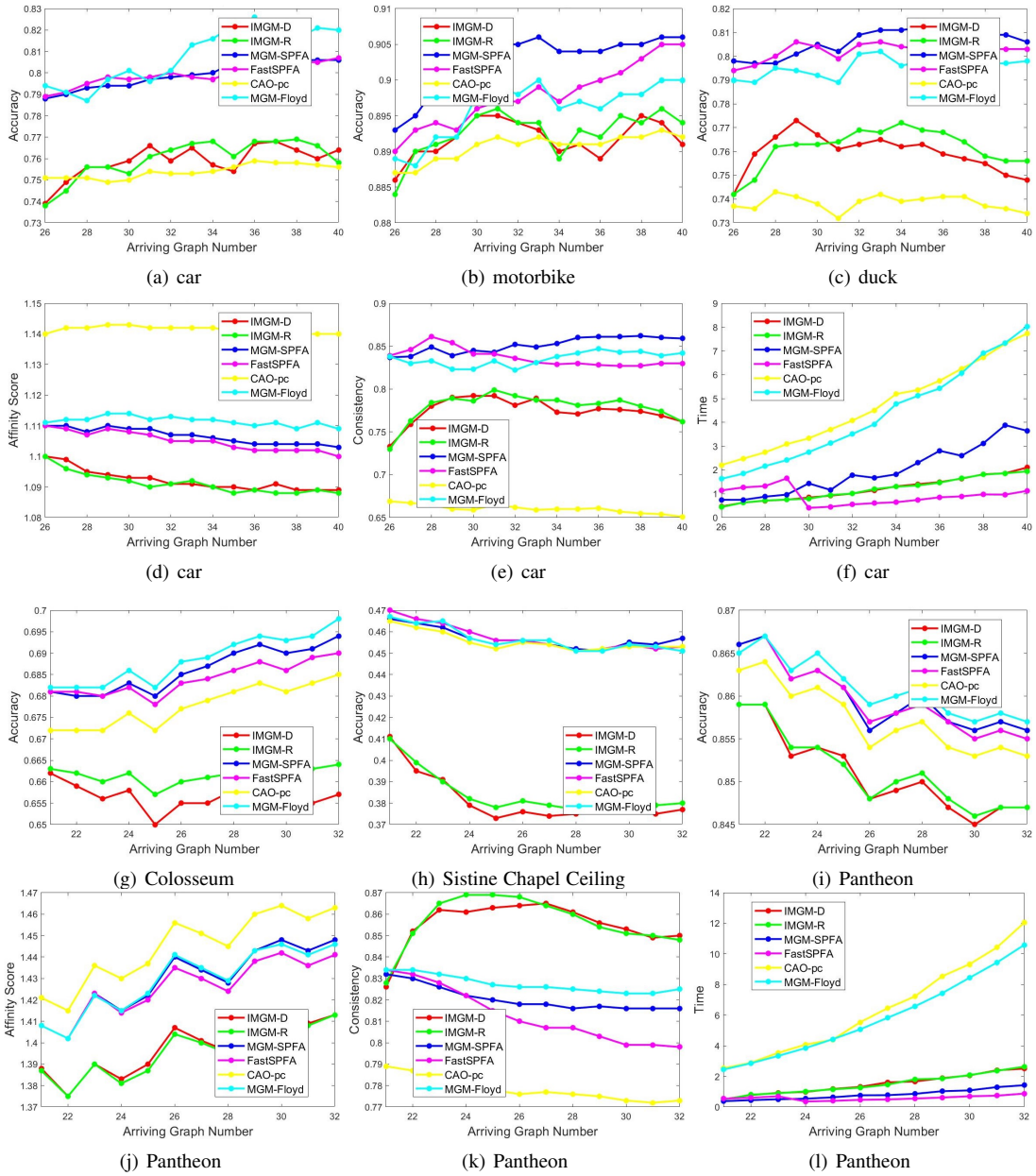


Fig. 6. Online incremental matching performance on Willow-ObjectClass and Sub-Rome16K. Both online and offline MGM methods are evaluated.

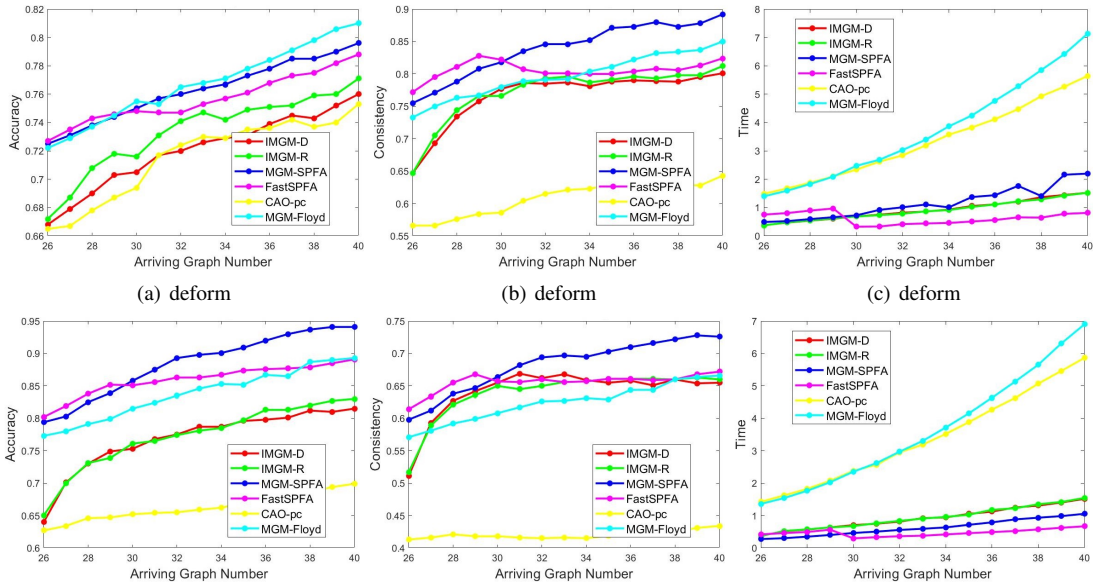
can be calculated by $\mathbf{K}_{ac;bd} = \exp(-\frac{(q_{ab}-q_{cd})^2}{\sigma^2})$, where σ^2 is the similarity sensitivity parameter. For outlier test, we follow the common inlier setting, whereby equal number of n_o outliers are added in each graph. As such, the matching matrix is always a permutation one which fits with our numerical studies. This protocol is also widely used in [4], [6], [18]. Note even the numbers of outliers are different among graphs in practice, one can introduce dummy nodes to make equal size of graphs [41].

Real image dataset. Two real image datasets are used.

Willow-ObjectClass [35] contains images from Caltech-256 and PASCAL VOC2007, which are categorized into five categories with quantity information as: 109 Face, 66 Winebottle, 50 Duck, 40 Car and 40 Motorbike. We choose the three categories except for Face and Winebottle as they are too easy for MGM. With 10 nodes manually labeled for each image, we additionally randomly generate n_o outliers for outlier test.

We also build another dataset as called Sub-Rome16K in this paper based on Rome16K [42] as it is originally designed for 3-D reconstruction rather than sparse graph matching. We collect three categories with different matching difficulty: Colosseum, Sistine Chapel Ceiling, Pantheon from Rome16K. Here we use the absolute matching accuracy over all the methods to illustrate the difficulty level of the class. Lower average matching accuracy denotes higher matching difficulty, and vice versa. In the experiment, we find Sistine Chapel Ceiling is the hardest one, and Pantheon is very easy to match while Colosseum is in middle. For each category, we select 32 graphs so that the intersection of key points in these graphs contains 30 points. We first choose n_i keypoints as the inliers, and then we randomly choose n_o keypoints from the rest $30 - n_i$ keypoints for each graph to be outliers.

In most experiments, we only use node coordinates to calculate second order affinity matrix, without considering the node-wise features. The adjacency matrix is constructed by sparse delaunay triangulation. We set the affinity matrix re-weighted by $\beta \in [0, 1]$ for both length and angle affinity:$K_{ia,jb} = \beta K^{len}_{ia,jb} + (1 - \beta)K^{ang}_{ia,jb}$. However, in Fig. 3, we also apply the same way



(a) deform (b) deform (c) deform

(d) deform+outlier (e) deform+outlier (f) deform+outlier

Fig. 7. Online matching performance on synthetic data (deform, deform+outlier). Offline MGM methods are also tested for efficiency comparison.

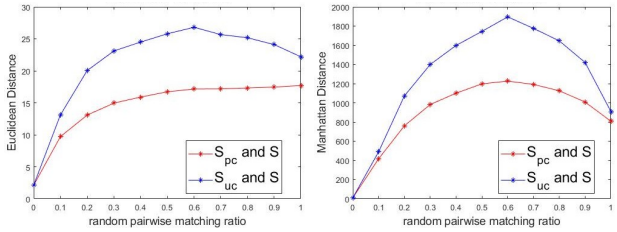


Fig. 8. Distance from S to S_{uc} and S to S_{pc}. On ground truth $\mathbb{X}$, we replace part of the matchings with random permutation matrix by varying ratio (x-axis) to generate different S and the corresponding S_{uc}, S_{pc}. It is shown S_{pc} is a better delegator to S.

TABLE 1
Setting for Tab. 2(top), Fig. 4, 5, 6, 7 ,9, 10, 12(middle) and 11(base).

ObjectClass	$n_i = 10, n_o = 0, \epsilon = 0, \rho = 0, \sigma^2 = 0.03, \beta = 0.9$
deform	$n_i = 10, n_o = 0, \epsilon = 0.15, \rho = 0, \sigma^2 = 0.05$
outlier	$n_i = 6, n_o = 4, \epsilon = 0, \rho = 0, \sigma^2 = 0.05$
deform+outlier	$n_i = 6, n_o = 4, \epsilon = 0.04, \rho = 0, \sigma^2 = 0.05$
mask	$n_i = 10, n_o = 0, \epsilon = 0.05, \rho = 0.9, \sigma^2 = 0.05$
ObjectClass	$n_i = 10, n_o = 2, \epsilon = 0, \rho = 0, \sigma^2 = 0.05, \beta = 0.9$
deform robustness	$n_i = 8, n_o = 2, \epsilon = 0.04 : 0.12, \rho = 0, \sigma^2 = 0.05$
outlier robustness	$n_i = 10, n_o = 6 : 12, \epsilon = 0.04, \rho = 0, \sigma^2 = 0.05$

with [36] to calculate higher-order affinity. When compared to peer methods e.g. [5], [20], [21], [22] (in Table 2), we follow all the protocol in [5]. Specifically, we adopt the deep features extracted from AlexNet [43] pretrained on ImageNet as node-wise first-order feature without accounting for the second-order or higher-order geometric features. The initial pairwise matchings are obtained by the linear assignment solver Hungarian method [44] and then fed into the MGM solvers.

4.3 Results and discussion

1) Offline comparison. Fig. 4 and Fig. 5 show the performance by offline methods on synthetic and real-world datasets. We compare Floyd with CAO in all kinds of evaluation metrics, where PC, UC, C-based methods taking S_{pc}, S_{uc}, S as score function respectively in Alg. 1. As we can see, Floyd-based methods outperform CAO-based methods with PC and UC approximation and achieve nearly the same accuracy as S score function with less time cost. In general, Floyd has a significant improvement over CAO on the consistency with a little cost of affinity score, e.g., the consistency of Floyd-PC increases about 0.20 while the affinity score drops only 0.03 on the synthetic deform dataset in Fig. 4(d) and 4(e). Floyd, compared with CAO, uses less comparison cost to find a relatively accurate composition path. In UC and PC-based methods, this allows a more precise consistency approximation via more updates of consistency matrix under the same time cost. In other words, there are T times update of consistency matrix in CAO and N times in Floyd, where T, the number of iteration rounds, is far less than the number of total graphs N. As for Floyd-C and CAO-C, less comparison means the former has less time cost than the latter, as shown Fig. 4(f), 5(f) and 5(l).

It is also worth noting that Floyd-PC has a significant speedup i.e. $1.5\times$ or even more with only 1% accuracy loss compared with Floyd-C and CAO-C. Besides, we find pairwise consistency is in general more effective than unary consistency, as shown in Fig. 8.

In addition with CAO based methods, we also compare our MGM-Floyd with four recent MGM (sometimes a.k.a. multi-way) mathing methods [5], [20], [21], [22] on Willow-ObjectClass. As shown in Table 2, Floyd based method shows its superiority on accuracy. On Motorbike, Floyd-PC even outperforms 17% over the best algorithm [5] among peer methods. Taking both accuracy and time into consideration, Floyd-PC outperforms consistently across all datasets. Compared with CAO-PC, it has 1% to 6% improvement in accuracy on real datasets and even 8% to 9% on the synthetic one. These results suggest the advantage of our discrete method as discussed in Sec. 3.5.

2) Online comparison. We compare four online methods: MGM-SPFA, FastSPFA, IMGM-D, IMGM-R, supplied with two offline methods MGM-Floyd and CAO-PC for time and accuracy. Since the peer methods IMGM-D, IMGM-R in [6] use pairwise consistency as the approximation of exact consistency, we also follow this setting in our MGM-SPFA, FastSPFA and MGM-Floyd. We set the least number of graphs for each cluster C_{min} to be 15 for synthetic dataset and Willow-ObjectClass, and 12 on

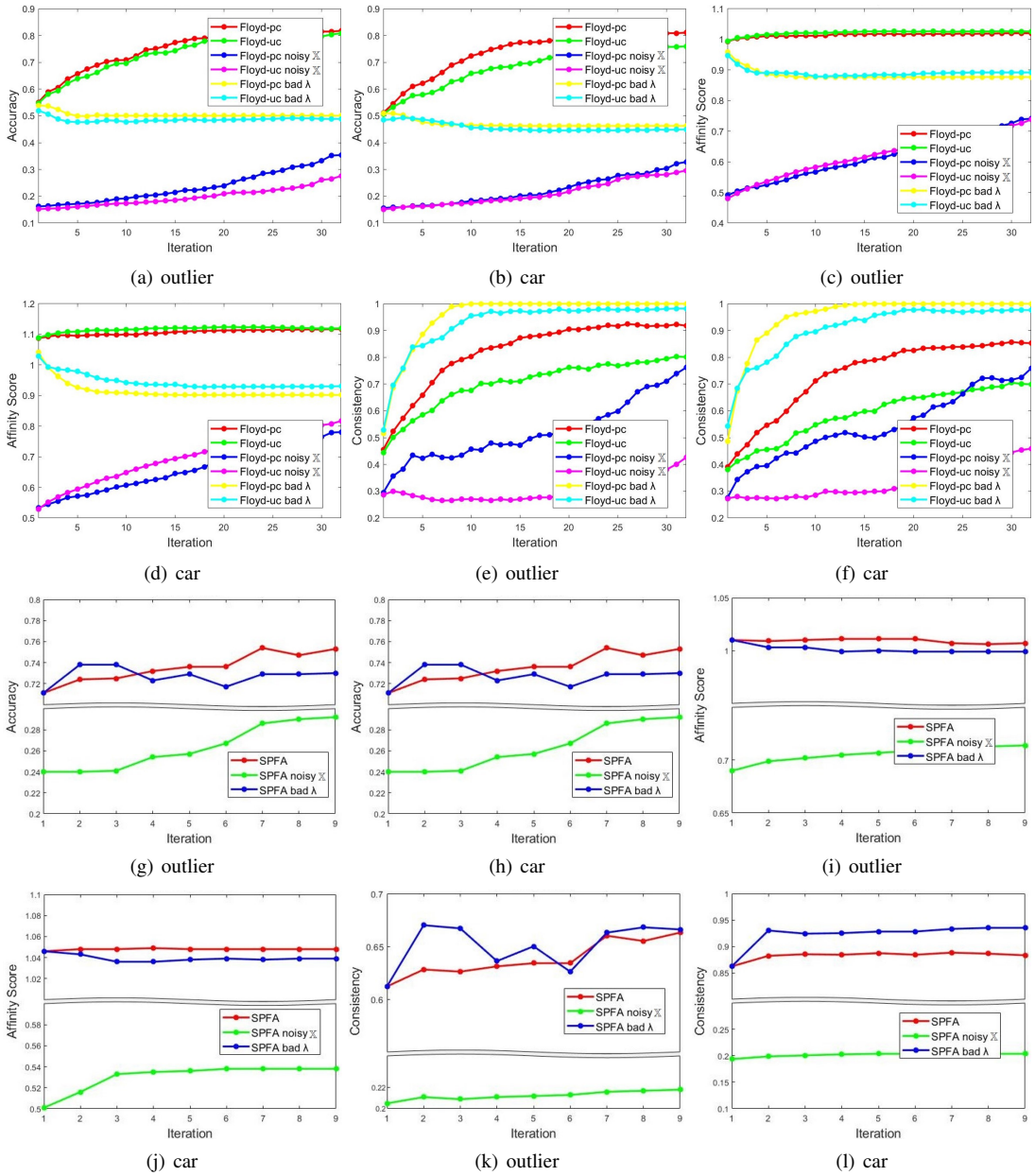


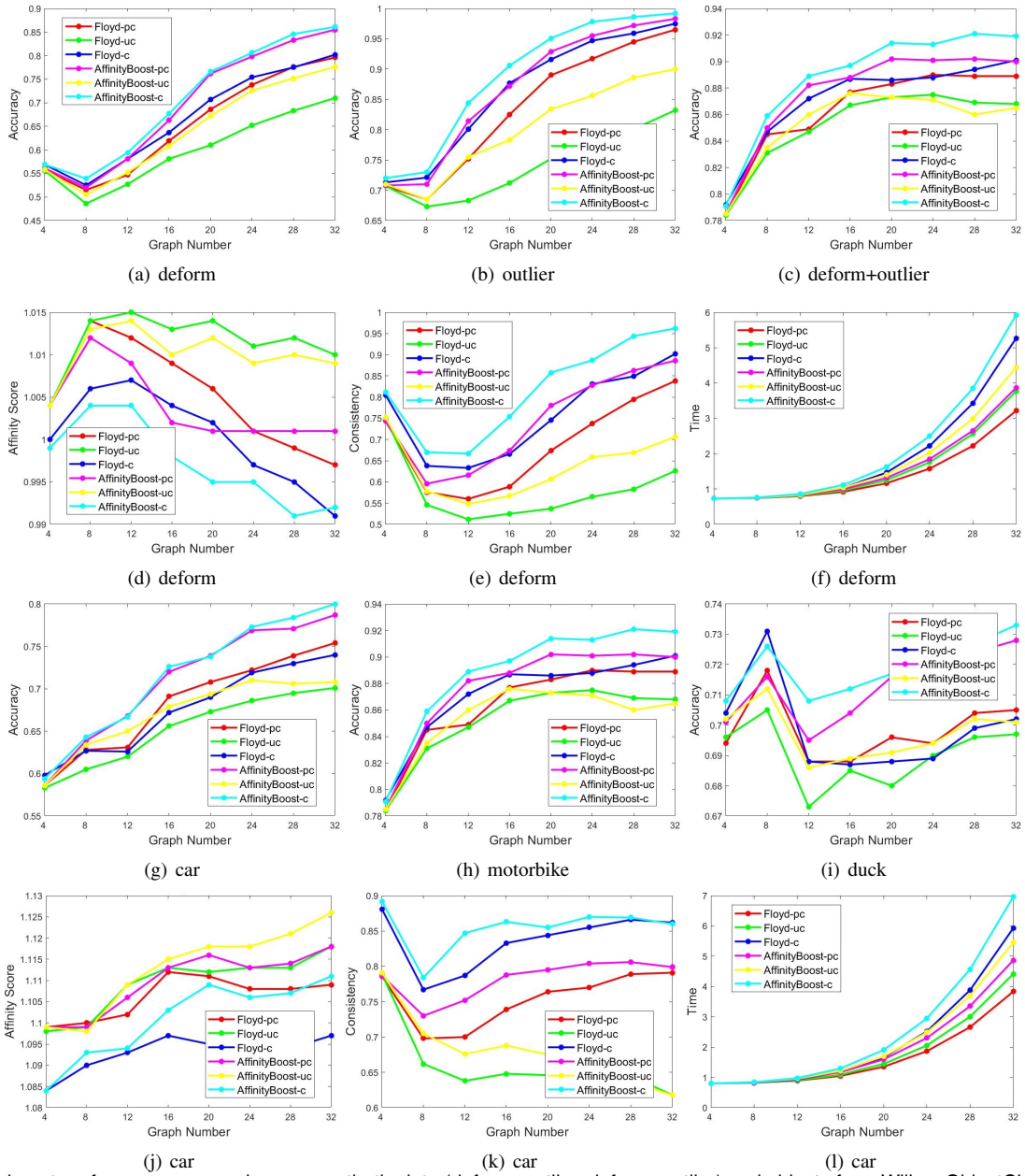
Fig. 9. MGM-Floyd (top two) and MGM-SPFA (bottom two) over iterations on synthetic outlier data and category 'car' from Willow-ObjectClass.

TABLE 2
Accuracy and time (in seconds) of the state-of-the-art offline MGM algorithms on Willow-ObjectClass. Best results of MGM methods are in bold. Note to obtain the two-graph matchings, here Hungarian method [44] is used whose input is the node-wise CNN features. In another word, no edge information is used due to the limitation of compared methods like MatchALS only accepts first-order feature but not second-order ones.

category	Hungarian	Floyd-pc	Floyd-uc	Floyd-c	Spectral [20]	MatchLift [21]	MatchALS [22]	MISM [5]
Car	0.503	0.844	0.840	**0.850**	0.601	0.665	0.629	0.750
Duck	0.442	**0.803**	0.800	0.793	0.485	0.554	0.525	0.732
Motorbike	0.317	0.821	0.817	**0.843**	0.255	0.296	0.310	0.653
Face	0.854	**1.000**	**1.000**	**1.000**	0.927	0.931	0.934	0.937
Winebottle	0.543	**0.934**	0.930	0.931	0.630	0.700	0.669	0.814
Time (Car)	1.263	9.377	10.960	14.981	**1.469**	17.695	2.622	3.966

sub-Rome16k (since the total number of graphs is only 32).

In Fig. 6 and Fig. 7, we can see our proposed online MGM-SPFA has the highest accuracy among all the online methods: MGM-SPFA has 7% to 8% accuracy improvement in average across all datasets. In 'deform+outlier' test of synthetic dataset, MGM-SPFA even outperforms MGM-Floyd about 5% in accuracy in Fig. 7(d). FastSPFA, using graph clustering for speedup, enjoys the fastest speed and a comparable accuracy among all online methods. It is worth noting that in Fig. 7(c), 7(f) and 6(f), the time of FastSPFA drops suddenly because of the introduction of a new cluster. This results in $2\times$ to $4\times$ speedup with only 1% to 2% accuracy loss on Willow-Object dataset. Comparing with IMGM-



(a) deform (b) outlier (c) deform+outlier

(d) deform (e) deform (f) deform

(g) car (h) motorbike (i) duck

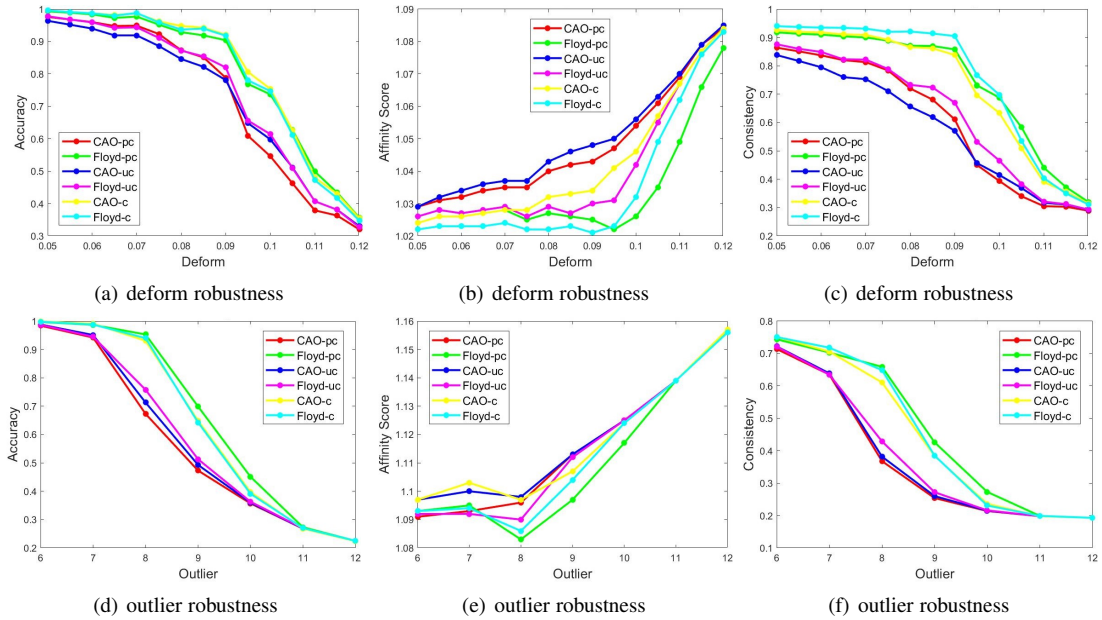
(j) car (k) car (l) car

Fig. 10. Affinity boost performance comparison on synthetic data (deform, outlier, deform+outlier) and objects from Willow-ObjectClass.

based methods, SPFA and FastSFPA outperform in both affinity score and consistency, thus leading a higher accuracy over them.

3) Effectiveness analysis. As shown in Fig. 4(c), Floyd-base methods have a lower accuracy compared with CAO-based methods in mask test where a bad initial matching is given. This mainly results from that the score function S tends to perform well given proper λ and good initial $\mathbb{X}$, which is mentioned in Assump. 1. We further use Fig. 9 to show the behavior of our methods, where we disable the affinity boost technique. For MGM-Floyd, when the initial accuracy is over 50% and λ is set to a proper value as 0.3, there is a rapid and steady improvement on accuracy, which almost increases 46% on synthetic data and 30% on real data. However, if the initial matching accuracy drops to 15%, accuracy increases extremely slow. In fact, in our tests we mimic such a low initial accuracy by intentionally replacing 92% initial pairwise matchings with random permutation matrix to reduce the accuracy of $\mathbb{X}$. It mainly attributes to the probability of Prop. 1 dropping to 60%, with the result that Assump. 1 does not hold anymore. Meanwhile, if we change λ to 0.8, we can see that the accuracy almost stays the same or even drops after iterations on both synthetic and real datasets. For MGM-SPFA, accuracy shows a similar tendency but less variance over iterations. As shown in Fig. 9(g) and Fig. 9(h), the accuracy improvements are only 4% and 2%. This is mainly caused by fewer iterations involved in the updating process and higher initial accuracy.

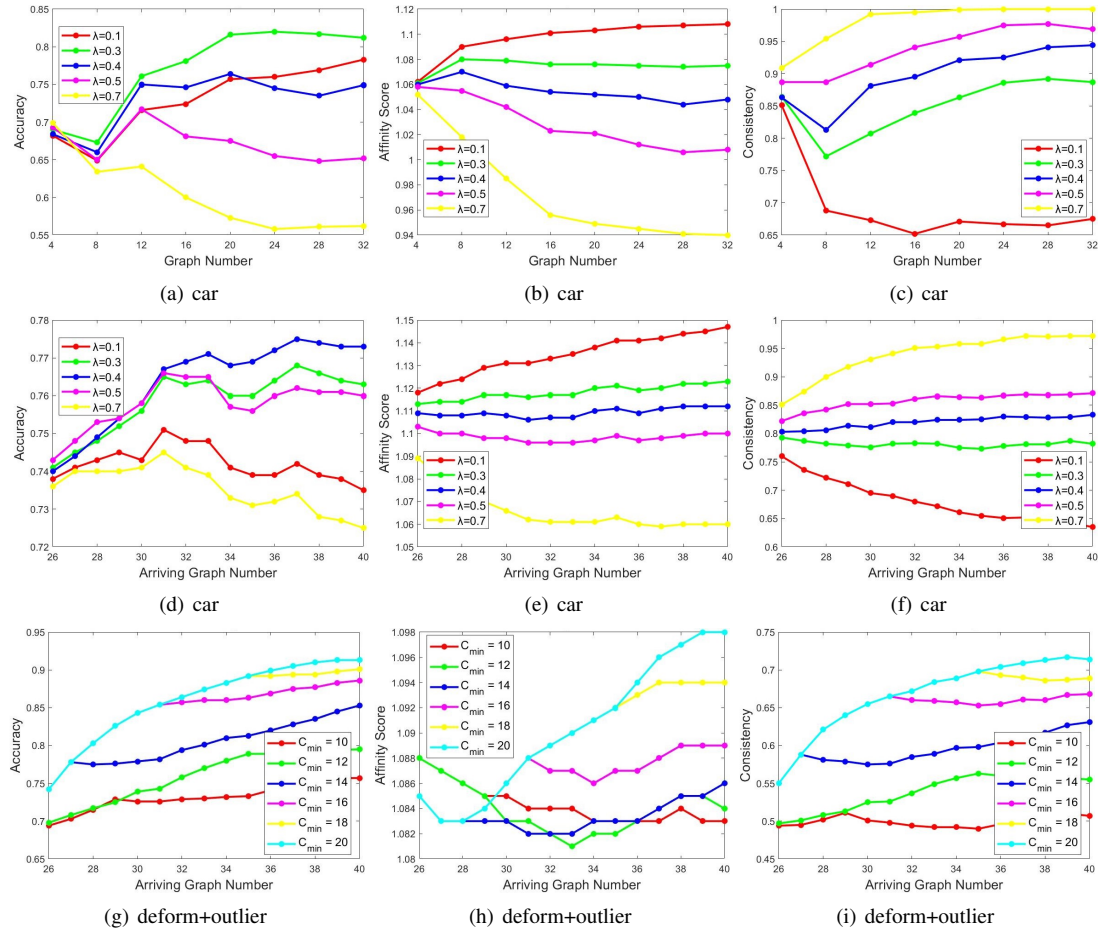
4) Affinity boost performance. In Sec. 3.2, a technique called 'affinity boost' is mentioned and Fig. 10 further illustrates the its effect. As one can see, affinity boosting improves the accuracy of all Floyd-base methods (Floyd-C, Floyd-PC, Floyd-UC) on both synthetic and real datasets. There is a significant accuracy boost on synthetic dataset by using it, i.e. the methods with affinity boost outperform others about 6% in deform test and 8% in deform+outlier test. While on real images, the improvements on accuracy are not so significant but still notable, i.e. the accuracy



(a) deform robustness (b) deform robustness (c) deform robustness

(d) outlier robustness (e) outlier robustness (f) outlier robustness

Fig. 11. Robustness test of offline methods over deformation, outliers. Due to noise, ground truth's affinity (equals to 1) is not the highest.



(a) car (b) car (c) car

(d) car (e) car (f) car

(g) deform+outlier (h) deform+outlier (i) deform+outlier

Fig. 12. Sensitivity test of weight parameter λ in Eq. 5 for MGM-Floyd (top), MGM-SPFA (middle) and C_{min} (i.e. the least number of graphs in a cluster) for FastSPFA (bottom) on synthetic data and category 'car' from Willow-ObjectClass.

of all Floyd-base method increases 2% on Duck dataset. The time cost is slightly increased by affinity boost, which is acceptable for the improvements on accuracy.

5) Robustness against noise. Fig. 11 verifies the robustness of our methods in the presence of deform and outliers. For robustness test involving both deform and outliers, our Floyd performs better than CAO on accuracy and consistency, while Floyd-PC surpasses all the other methods. In deform test, nearly the same accuracy is achieved by CAO-C and Floyd-C, which outperform other approximation methods with 10% in accuracy

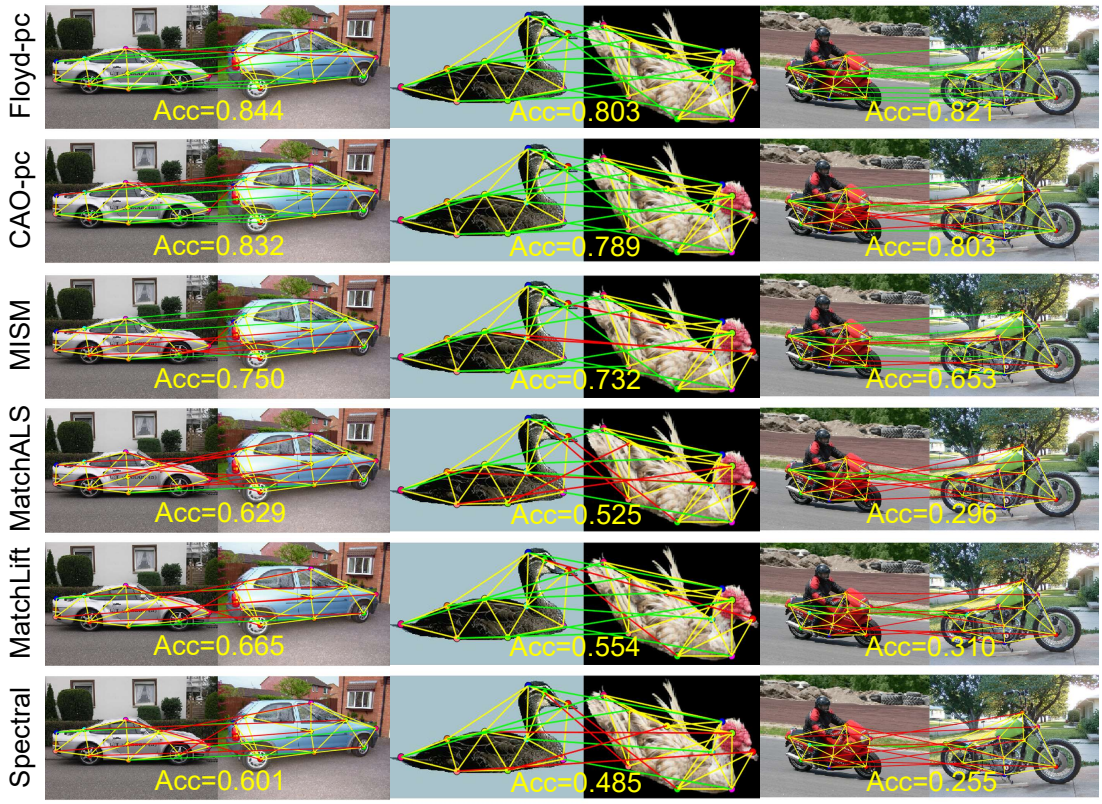


Fig. 13. Matchings (correct in green and otherwise red) of Car, Duck, Motorbike from Willow-ObjectClass. We compare [4], [5], [20], [21], [22].

for $\epsilon \in [0.09, 0.11]$. For outlier test, there is an even larger accuracy improvement: 5% over C-based methods and 10% to 20% over other approximation methods.

Moreover, we find that on both synthetic and real datasets to support our methods' robustness. As shown in Fig. 5 and Fig. 11, we can see that when the absolute matching accuracy is around 0.5, our algorithm surpasses other methods by a large extent while the advantage over others gradually drops when the matching accuracy becomes higher or lower. Therefore, our algorithm MGM-Floyd is more capable of tackling the complex case in graph matching scenarios. Besides, Fig. 6 also shows that MGM-SPFA and FastSPFA have similar property.

6) Sensitivity test of λ and cluster counts. Fig. 12 shows the behavior regarding different λ values. A balance between affinity score and consistency can lead to better performance, while extreme values e.g. $\lambda = 0.1$ or 0.7 cause degeneration on both synthetic and real datasets. MGM-SPFA also shows similar behavior while the performance is even more stable for $\lambda \in [0.3, 0.5]$. The accuracy variations of MGM-SPFA when $\lambda \in [0.3, 0.5]$ are 2% and 1% on synthetic and real datasets. The cluster counts experiment is conducted and the result is shown in Fig. 12. When C_{min} increases, the accuracy raises steadily, but the time cost is also going up. As illustrated in Fig. 12(h) and 12(i), this increment of accuracy results from the gradual improvement of consistency, while the affinity score stays nearly the same.

4.4 Concluding remarks

We draw a few concluding remarks as follows:

1) For offline setting, MGM-Floyd outperforms CAO [4] on both synthetic and real datasets in accuracy and consistency, with similar time cost. It further notably outperforms recent MGM algorithms [5], [20], [21], [22] in accuracy, though being less efficient than these continuous relaxation methods; 2) For online setting, the proposed MGM-SPFA outperforms compared methods IMGM [6] on both synthetic and real datasets in accuracy and slightly more time overhead. Moreover, our speedup version FastSPFA further outperforms IMGM [6] in accuracy, consistency and efficiency; 3) The experiments well support our numerical analysis i.e. Assump. 1, based on which our methods are devised. When the accuracy of initial matchings $\mathbb{X}$ is too low e.g. less than 10%, the improvement by iterations of our methods can be more restricted than CAO [4], or when the weight parameter is intentionally set too high e.g. $\lambda = 0.8$. Note methods [4], [6] will also fail given large λ as already shown in [4]. Fortunately, these settings are hardly possible since the employing off-the-shelf two-graph matching solver e.g. RRWM, RRWHM [8], [36] can easily obtain good initial $\mathbb{X}$ and our sensitivity study has shown the robustness of our methods to λ, different two-graph matching solvers, and scale of nodes for graphs to be matched.

5 Conclusion

This paper studies the problem of finding optimal composition chain on the supergraph to generate new matchings for MGM. By a careful numerical study on the behavior of composition chain for both affinity score and consistency, we formulate the offline and online MGM as the task of finding all pairwise shortest paths and single-source shortest paths on supergraph respectively. The devised methods correspond to Floyd method and SPFA being more effective than ad-hoc composition methods in [4], [6]. We also develop the speedup version FastSPFA for online MGM by graph clustering. Superior performance is achieved on synthetic and real-world data against peer methods [5], [20], [21], [22].

ACKNOWLEDGEMENTS

The work is partially supported by China Major State Research Development Program (2018AAA0100704), NSFC (61972250, U19B2035) and the Open Project Program of the National Laboratory of Pattern Recognition (NLPR). We thank the anonymous reviewers for their valuable comments for improving the paper.

REFERENCES

[1] R. W. Floyd, "Algorithm 97: shortest path," *Communications of the ACM*, vol. 5, no. 6, p. 345, 1962.

[2] E. F. Moore, "The shortest path through a maze," in *Proc. Int. Symp. Switching Theory, 1959*, 1959, pp. 285–292.

[3] F. Duan, "A faster algorithm for shortest-path spfa [j]," *Journal of Southwest Jiaotong University*, vol. 29, no. 2, 1994.

[4] J. Yan, M. Cho, H. Zha, X. Yang, and S. M. Chu, "Multi-graph matching via affinity optimization with graduated consistency regularization," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 6, pp. 1228–1242, 2016.

[5] Q. Wang, X. Zhou, and K. Daniilidis, "Multi-image semantic matching by mining consistent features," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 685–694.

[6] T. Yu, J. Yan, W. Liu, and B. Li, "Incremental multi-graph matching via diversity and randomness based graph clustering," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 139–154.

[7] Z. Zhang, "Iterative point matching for registration of free-form curves and surfaces," *International journal of computer vision*, vol. 13, no. 2, pp. 119–152, 1994.

[8] M. Cho, J. Lee, and K. M. Lee, "Reweighted random walks for graph matching," in *European conference on Computer vision*. Springer, 2010, pp. 492–505.

[9] J. Yan, C. Zhang, H. Zha, W. Liu, X. Yang, and S. M. Chu, "Discrete hyper-graph matching," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1520–1528.

[10] Z.-Y. Liu, H. Qiao, X. Yang, and S. C. Hoi, "Graph matching by simplified convex-concave relaxation procedure," *International Journal of Computer Vision*, vol. 109, no. 3, pp. 169–186, 2014.

[11] M. Leordeanu and M. Hebert, "A spectral technique for correspondence problems using pairwise constraints," in *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, vol. 2. IEEE, 2005, pp. 1482–1489.

[12] M. Leordeanu, R. Sukthankar, and M. Hebert, "Unsupervised learning for graph matching," *International journal of computer vision*, vol. 96, no. 1, pp. 28–45, 2012.

[13] E. M. Loiola, N. M. M. de Abreu, P. O. Boaventura-Netto, P. Hahn, and T. Querido, "A survey for the quadratic assignment problem," *European journal of operational research*, vol. 176, no. 2, pp. 657–690, 2007.

[14] E. L. Lawler, "The quadratic assignment problem," *Management science*, vol. 9, no. 4, pp. 586–599, 1963.

[15] S. Gold and A. Rangarajan, "A graduated assignment algorithm for graph matching," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 18, no. 4, pp. 377–388, 1996.

[16] B. J. van Wyk and M. A. van Wyk, "A pocs-based graph matching algorithm," *IEEE transactions on pattern analysis and machine intelligence*, vol. 26, no. 11, pp. 1526–1530, 2004.

[17] F. Fraundorfer and D. Scaramuzza, "Visual odometry: Part ii: Matching, robustness, optimization, and applications," *IEEE Robotics & Automation Magazine*, vol. 19, no. 2, pp. 78–90, 2012.

[18] J. Yan, Y. Tian, H. Zha, X. Yang, Y. Zhang, and S. M. Chu, "Joint optimization for consistent multiple graph matching," in *Proceedings of the IEEE International Conference on Computer Vision*, 2013, pp. 1649–1656.

[19] Q.-X. Huang and L. Guibas, "Consistent shape maps via semidefinite programming," in *Computer Graphics Forum*, vol. 32, no. 5. Wiley Online Library, 2013, pp. 177–186.

[20] D. Pachauri, R. Kondor, and V. Singh, "Solving the multi-way matching problem by permutation synchronization," in *Advances in neural information processing systems*, 2013, pp. 1860–1868.

[21] Y. Chen, L. Guibas, and Q. Huang, "Near-optimal joint object matching via convex relaxation," in *International Conference on Machine Learning*, 2014, pp. 100–108.

[22] X. Zhou, M. Zhu, and K. Daniilidis, "Multi-image matching via fast alternating minimization," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 4032–4040.

[23] X. Shi, H. Ling, W. Hu, J. Xing, and Y. Zhang, "Tensor power iteration for multi-graph matching," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 5062–5070.

[24] S. Leonardos, X. Zhou, and K. Daniilidis, "Distributed consistent data association via permutation synchronization," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 2645–2652.

[25] N. Hu, Q. Huang, B. Thibert, and L. J. Guibas, "Distributable consistent multi-object matching," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 2463–2471.

[26] M. R. Henzinger, P. Klein, S. Rao, and S. Subramanian, "Faster shortest-path algorithms for planar graphs," *journal of computer and system sciences*, vol. 55, no. 1, pp. 3–23, 1997.

[27] J. Yan, X.-C. Yin, W. Lin, C. Deng, H. Zha, and X. Yang, "A short survey of recent advances in graph matching," in *Proceedings of the 2016 ACM on International Conference on Multimedia Retrieval*, 2016, pp. 167–174.

[28] J. Yan, J. Wang, H. Zha, X. Yang, and S. Chu, "Consistency-driven alternating optimization for multigraph matching: A unified approach," *IEEE Transactions on Image Processing*, vol. 24, no. 3, 2015.

[29] F. Bernard, J. Thunberg, J. Goncalves, and C. Theobalt, "Synchronisation of partial multi-matchings via non-negative factorisations," *Pattern Recognition*, vol. 92, pp. 146–155, 2019.

[30] T. Birdal and U. Simsekli, "Probabilistic permutation synchronization using the riemannian structure of the birkhoff polytope," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 11 097–11 108.

[31] S. Leonardos and K. Daniilidis, "A distributed optimization approach to consistent multiway matching," in *2018 IEEE Conference on Decision and Control (CDC)*. IEEE, 2018, pp. 89–96.

[32] V. G. Kim, W. Li, N. J. Mitra, S. DiVerdi, and T. A. Funkhouser, "Exploring collections of 3d models using fuzzy correspondences," *ACM Transactions on Graphics*, vol. 31, no. 4, p. 54, 2012.

[33] P. Swoboda, D. Kainm"uller, A. Mokarian, C. Theobalt, and F. Bernard, "A convex relaxation for multi-graph matching," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[34] A. Chakraborty, A. Das, and A. K. Roy-Chowdhury, "Network consistent data association," *IEEE transactions on pattern analysis and machine intelligence*, vol. 38, no. 9, pp. 1859–1871, 2016.

[35] M. Cho, K. Alahari, and J. Ponce, "Learning graphs to match," in *2013 IEEE International Conference on Computer Vision*, 2013, pp. 25–32.

[36] J. Lee, M. Cho, and K. M. Lee, "Hyper-graph matching via reweighted random walks," in *CVPR 2011*, 2011, pp. 1633–1640.

[37] J. Yan, H. Xu, H. Zha, X. Yang, H. Liu, and S. Chu, "A matrix decomposition perspective to multiple graph matching," in *2015 IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 199–207.

[38] J. Yan, Zhe Ren, H. Zha, and S. Chu, "A constrained clustering based approach for matching a collection of feature sets," in *2016 23rd International Conference on Pattern Recognition (ICPR)*, 2016, pp. 3832–3837.

[39] R. Tron, X. Zhou, C. Esteves, and K. Daniilidis, "Fast multi-image matching via density-based clustering," in *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 4077–4086.

[40] M. Leordeanu, M. Hebert, and R. Sukthankar, "An integer projected fixed point method for graph matching and map inference," in *Advances in Neural Information Processing Systems 22*, Y. Bengio, D. Schuurmans, J. D. Lafferty, C. K. I. Williams, and A. Culotta, Eds. Curran Associates, Inc., 2009, pp. 1114–1122.

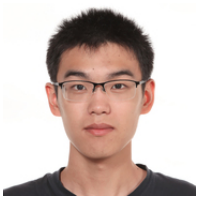
[41] F. Zhou and F. De la Torre, "Factorized graph matching," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 127–134.

[42] Y. Li, N. Snavely, and D. P. Huttenlocher, "Location recognition using prioritized feature matching," in *European conference on computer vision*. Springer, 2010, pp. 791–804.

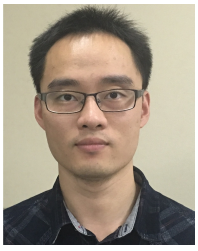
[43] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[44] H. W. Kuhn, "The hungarian method for the assignment problem," in *Export. Naval Research Logistics Quarterly*, 1955, pp. 83–97.

Zetian Jiang is currently an Undergraduate Student with ACM Class in Zhiyuan College (Honored Program), Shanghai Jiao Tong University. He also worked as a visiting student in Computer Science Department, Stanford University. His research interests include machine learning and computer vision.



Tianzhe Wang is currently an Undergraduate Student with ACM Class in Zhiyuan College (Honored Program), Shanghai Jiao Tong University. He won gold medals in ACM-ICPC Regional Contest Qingdao Site and Chinese Collegiate Programming Contest. He was the winner of Visual Wake Words Challenge@CVPR 2019 and ranked 3-rd place(1-st in academia) in Low Power Image Recognition Challenge@CVPR 2019. He conducted research as a visiting student in Massachusetts Institute of Technology. His research interests include machine learning, theoretical computing.



Junchi Yan (M'10) is currently an Associate Professor with Department of Computer Science and Engineering, and MoE Key Lab of Artificial Intelligence, AI Institute, Shanghai Jiao Tong University, China. Before that, he was a Senior Research Staff Member and Principal Scientist with IBM Research where he started his career in April 2011. He obtained the Ph.D. degree in Electrical Engineering from Shanghai Jiao Tong University. He received the ACM China Doctoral Dissertation Nomination Award and China Computer Federation Doctoral Dissertation Award. His research interests include machine learning and computer vision. He serves as Associate Editor for IEEE ACCESS, Senior PC for CIKM 2019 and Area Chair for ICPR 2020, CVPR 2021. He also served (Managing) Guest Editor for IEEE Transactions on Neural Networks and Learning Systems, Pattern Recognition Letters, and Pattern Recognition. He is a member of IEEE.