Progressive Graph Matching: Making a Move of Graphs via Probabilistic Voting

Minsu Cho
Kyoung Mu Lee

Proc. Computer Vision and Pattern Recognition (CVPR) 2012

Graph Matching

Given two attributed graphs $\bar{G}^P = (\bar{V}^P, \bar{E}^P, \bar{A}^P)$ and $\bar{G}^Q = (\bar{V}^Q, \bar{E}^Q, \bar{A}^Q)$ with n^P and n^Q nodes respectively.

A result of graph matching is a subset of possible correspondences between those graphs, which can be represented in form of assignment matrix $X \in \{0,1\}^{n^P \times n^Q}$:

$$X_{ia} = \begin{cases} 1 & \text{node } v_i \in \bar{V}^P \text{matches } v_a \in \bar{V}^Q \\ 0 & \text{otherwise} \end{cases}$$

General formulation:

$$x^* = \arg\max S(x)$$

$$s.t. \begin{cases} x \in 0,1^{n^P n^Q} \\ \sum_{i=1}^{n^P} x_{ia} \leq 1 \\ \sum_{a=1}^{n^Q} x_{ia} \leq 1 \end{cases}$$

The objective function $S(x)$ measures the similarity between the graph attributes.

Integer Quadratic Programming

Consider two types of similarity: node and edge similarity. Those can be combined in an affinity matrix W, where non-diagonal elements $W_{ia,jb}$ represent edge similarity of the edges $e_{ij} \in \bar{E}^P$ and $e_{ab} \in \bar{E}^Q$ and diagonal elements $W_{ia,ia}$ represent similarity between nodes $v_i \in \bar{V}^P$ and $v_a \in \bar{V}^Q$.
That leads to the integer quadratic optimization problem:

$$x^* = \arg\max x^T W x$$

$$\text{s.t.} \begin{cases} x \in 0, 1^{n^P n^Q} \\ \sum_{i=1}^{n^P} x_{ia} \leq 1 \\ \sum_{a=1}^{n^Q} x_{ia} \leq 1 \end{cases}$$

Issues

- Quadratic assignemnt problem is NP-complete
- Computation of W for a large graph is almost infeasible
- Reduction of the complexity often leads to worse matching results

Progressive Graph Matching

Refer to the initial graphs $\bar{G}^P$ and $\bar{G}^Q$ as maximal graphs. For each maximal graph a subgraph induced by a reduced set of nodes is called active graph (G^P and G^Q respectively).

Idea: match maximal graphs by iteratively matching their active graphs ($t = 0, 1, \dots$)

- Graph Matching
  match current active graphs $G_t^P = (V_t^P, E_t^P, A_t^P)$ and $G_t^Q = (V_t^Q, E_t^Q, A_t^Q)$, $G_t^P \subset \bar{G}^P, G_t^Q \subset \bar{G}^Q$.
- Graph Progression
  update active graphs to improve matching score on the next matching step

Graph Matching

To reduce the complexity of graph matching consider a given set of candidate matches $C_t \subset V^P \times V^Q$.

The corresponding active graphs are defined by the nodes appeared in C_t.

This method reduces the size of active graphs and makes a affinity matrix more sparse.

Graph Progression I

Let $M_t = \{m_1, \ldots, m_{|M_t|}\} \subset C_t$ with $m_i = (v^P_{p_i}, v^Q_{q_i})$ be a result of graph matching of two active graphs and s_t it's score.
Consider conditional joint probability $p(V^P, V^Q|M_t)$:

$$\begin{aligned} p(V^P, V^Q|M_t) &= \sum_{m_i \in M_t} p(V^P, V^Q, M = m_i|M_t) \\ &= \sum_{m_i \in M_t} p(V^Q|V^P, M = m_i, M_t) \\ &\qquad p(V^P|M = m_i, M_t)p(M = m_i|M_t) \end{aligned}$$

Graph Progression II

$p(v_i^P, v_a^Q|M_t)$ conditional probability of match (v_i^P, v_a^Q) between two maximal graphs.
Base on the probability distribution $p(V^P, V^Q|M_t)$ new candidate set C_{t+1} consists of N_c matches best matches.
Given new candidate set C_{t+1} we obtain new active graphs G_{t+1}^P and G_{t+1}^P.
The condition $M_t \subset C_{t+1}$ ensures that $s_{t+1} \geq s_t$, if M_t is an optimal matching.

Algorithm

Algorithmus 1 progressiveGraphMatching($\bar{G}^P$, $\bar{G}^Q$, N_c) [1]

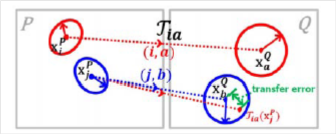
$C_0 = Find\_Initial\_Candidates(\bar{G}^P, \bar{G}^Q, N_c)$
$t = 0, s_0 = 0$
while score increases **do**
  $(M_t, s_t) = Graph\_Matching(C_t)$
  $p(v_p^P, v_q^Q | M_t) = 0 \;\; \forall v_p^P \in \bar{V}^P, v_q^Q \in \bar{V}^Q$
  for each $m_i = (v_p^P, v_q^Q) \in M_t$ **do**
    $N_A = \{v^P \in \bar{V}^Q | p(v^P | m_i, M_t) > \epsilon\}$
    for each $v_j^P \in N_A$ **do**
      $N_B = \{v^Q \in \bar{V}^Q | p(v^Q | v_j^P, m_i, M_t) > \epsilon\}$
      for each $v_b^Q \in N_b$ **do**
        $p(v_p^P, v_q^Q | M_t) = p(v_p^P, v_q^Q | M_t) + p(v_b^Q | v_j^P, m_i, M_t) p(v_j^P | m_i, M_t) p(m_i | M_t)$
      end for
    end for
  end for
  $C_{t+1} = N_c$ best matches based on $p(V^P, V^Q | M_t)$, which contains M_t
  $t = t + 1$
end while
return M_t

Matching of images I

Given two images.

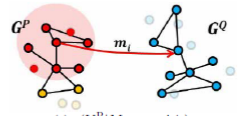
- Detected features of the images represent nodes of the maximal graphs (and their geometric relations as edges)
- Node similarity = similarity of feature descriptors of two graphs
- Edge similarity = Symmetric Transfer Error (STE) (see [2])

Derive affine homography transformation $\tau_{ia}(\cdot)$ between two features
The transfer error of (j, b) with respect to (i, a)
$d_{jb|ia} = \|x_b^Q - \tau_{ia}(x_j^P)\|$
Edge similarity $W_{ia,jb} =$
$\max(0, \alpha - \frac{d_{jb|ia} + d_{bj|ai} + d_{ia|jb} + d_{ai|jb}}{4})$



Matching of images II

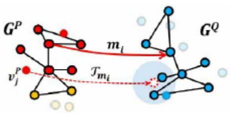
- $p(M = m_i | M_t) = score(m_i) / \sum_i score(m_i)$
- Assuming $m_i = (v^P_{p_i}, v^Q_{q_i})$ and use function $kNN(\cdot, k)$ to find k-nearest neighbors of a node



(a) $p(V^P|M = \mathbf{m}_i, \mathcal{M}_t)$

$$p(V^P = v^P_j | M = m_i, M_t) = \begin{cases} 1/k_1, & \text{if } v^P_j \in kNN(v^P_{p_i}, k_1) \\ 0, & \text{otherwise} \end{cases}$$

- Let $NN(\cdot)$ define a nearest neighbor of a node



(b) $p(V^Q|V^P = v^P_j, M = \mathbf{m}_i, \mathcal{M}_t)$

$$p(V^Q = v^Q_b | V^P = v^P_j, M = m_i, M_t) =$$

$$\begin{cases} 1, & \text{if } v^Q_b = NN(\tau_{m_i}(p^P_j)) \text{ and } (v^P_j, v^Q_b) \in M_t \\ \exp(-d_{jb|mi})/Z, & \text{if } v^Q_b \in kNN(\tau_{m_i}(p^P_j), k_2) \text{ and } (v^P_j, NN(\tau_{m_i}(p^P_j))) \notin M_t \\ 0, & \text{otherwise} \end{cases}$$

$Z = \sum_{v^Q_b \in kNN(\tau_{m_i}(p^P_j), k_2)} \exp(-d_{jb|mi})$ is a normalization constant

Evaluation

The End

Thank you for your attention!

References

[1] Minsu Cho and Kyoung Mu Lee. Progressive graph matching: Making a move of graphs via probabilistic voting. *Proc. Computer Vision and Pattern Recognition (CVPR)*, 2012.

[2] Minsu Cho, Kyoung Mu Lee, and Jungmin Lee. Reweighted random walks for graph matching. *European Conference on Computer Vision (ECCV)*, 2010.