



Graphical models for graph matching: Approximate models and optimal algorithms

Terry Caelli *, Tiberio Caetano

National ICT Australia, Canberra, ACT 0200, Australia

Received 5 May 2004; received in revised form 25 October 2004
Available online 18 December 2004

Abstract

Comparing scene, pattern or object models to structures in images or determining the correspondence between two point sets are examples of attributed graph matching. In this paper we show how such problems can be posed as one of inference over hidden Markov random fields. We review some well known inference methods studied over past decades and show how the Junction Tree framework from Graphical Models leads to algorithms that outperform traditional relaxation-based ones.

© 2004 Elsevier B.V. All rights reserved.

Keywords: Attributed graph matching; Hidden Markov random fields; Relaxation labeling; Junction Tree algorithms

In memoriam

It was the Summer of 1994, at a NATO workshop on Geometry and Vision and Professor Azriel Rosenfeld was there. He was sitting at the back of the lecture room vigorously reading the conference papers, listening to the speaker, updating his library references and participating in the discourse of the meeting—to me, that was Azriel. Endless energy, enthusiasm and always in the cen-

tre of things. This paper has been written to celebrate his life and contributions to Computer Vision which are as much broad as they are foundational. He uniquely qualifies to be a peer of our community and this special edition acknowledges that.

(Terry Caelli)

1. Introduction

The most common data model for relational structures are graphs, and the general problem of attributed graph matching has a long history in

* Corresponding author.

E-mail address: terry.caelli@rsise.anu.edu.au (T. Caelli).

computer vision, dating back to the work of Tsai and Fu (1979) and Fu (1983). In this paper, attributed graphs refer to graphs whose nodes (vertices) and edges have attributes. For example, for vision applications, vertices may have unary attributes of position, orientation, area, while edges may have binary attributes of lengths and angle differences. Over the past decade or more there has been a renewed interest in algorithms for matching such structures and approaches vary from graph theoretical (Bunke and Shearer, 1998) through to evolutionary biology approaches (Pelillo et al., 1999). However, recently, the vision and pattern recognition community has explored two new classes of solutions. The first is based on deterministic linear least squares (van Wyk et al., 2002) and graph eigenspace projections (Shapiro and Brady, 1992; Caelli and Kosinov, 2004) that fall into the class of kernel-based variational methods. The second, of interest in this paper, are those based on probabilistic models, including Probabilistic relaxation labeling (PRL), which has been used for graph matching for decades (Kittler and Hancock, 1989; Christmas et al., 1995). The key ideas behind any of these methods involve the assumption that “structure” is defined probabilistically for elements and their relations, and the identification or alignment of structures involves optimizing a matching likelihood function. In this way, we will examine how this general approach can be posed as one of inference over a hidden Markov random field (HMRF). HMRFs have been used in image encoding (Cheng and Bouman, 2001), segmentation (Zhang et al., 2001) and image understanding (Caelli et al., 2003). However, to this stage they have not been used for graph matching.

2. The basic HMRF graph model

Assume we have two attributed graphs, $G_s = (\mathcal{V}_s, \mathcal{E}_s)$ and $G_x = (\mathcal{V}_x, \mathcal{E}_x)$, representing a scene and a template/query, respectively. \mathcal{V}_s , \mathcal{E}_s , \mathcal{V}_x and \mathcal{E}_x represent the vector attributes of vertices and edges of the scene and template graphs, respectively. The scene consists of the template plus a set of “background” or “noisy” nodes so that both graphs do not, in general, have the same

numbers of vertices. A single node in the graph G_x is defined by x_i , and in the graph G_s by s_z . Each node in each graph has a unary attribute vector: y_s^α in G_s and y_x^j in G_x (with $1 \leq \alpha \leq S$ and $1 \leq j \leq T$, where S and T are the number of nodes in G_s and G_x , respectively). Binary attributes for each pair of nodes within a single graph are denoted by $y_s^{\alpha\beta}$ for G_s and by y_x^{ij} for G_x .

As we will see, the subgraph isomorphism problem can be posed as that of assigning to each x_i a unique label, s_z —assuming that there is one “signal” embedded in the “scene”. We define a HMRF over the template graph, G_x , by considering each node x_i in G_x as a random variable (rv) which can assume any of S possible values, s_z , corresponding to the nodes of G_s . The connectivity structure and the potential functions of the HMRF are specified in the following.

2.1. The observation component

Each x_i corresponds to a discrete site in the “hidden” layer of the HMRF. To each x_i we have the vertex (unary) observed attribute vector: y_x^j whose values, we will see, are dependent on the “states” (labels) of x_i corresponding to the vertices of G_s . That is, y_x^j is dependent only on its vertex x_i , as

$$B_i = p(y_x^j | x_i) = [(C_1(y_x^j, y_s^1), \dots, C_1(y_x^j, y_s^S))]^t$$

where each single element is given by $B_{iz} = p(y_x^j | x_i = s_z) = C_1(y_x^j, y_s^z)$, where y_s^z is the unary attribute associated with the state s_z of rv x_i . The unary compatibility function, $C_1(y_x^j, y_s^z)$, represents how likely should be the assignment $x_i = s_z$, and is defined by the multivariate gaussian

$$C_1(y_x^j, y_s^z) = \mathcal{N}(y_x^j; y_s^z, \text{cov})$$

where cov is the covariance matrix of the complete set of unary attributes in G_s . The notation $\mathcal{N}(y_x^j; y_s^z, \text{cov})$ refers to the value of the Gaussian function with mean vector y_s^z and covariance matrix cov for the particular argument y_x^j .

Consequently the conditional density function, $p(y_x^j | x_i)$, gives rise to a compatibility function $\phi_i(y_x^j, x_i)$ that defines, for each possible outcome of x_i , the compatibility between y_x^j and y_s^z . Rewriting, then, we have

$$\phi_i(y_x^i, x_i) = [(C_1(y_x^i, y_s^1), \dots, C_1(y_x^i, y_s^S))]^t$$

where subscript i in ϕ_i denotes that this vector is dependent solely on i . In the same way we can write the single compatibility coefficient $\phi_{ix} = C_1(y_x^i, y_s^s)$.

2.2. The Markov component

From the above, each unary evidence node in the HMRF (y_x^i) is dependent on its hidden node, x_i , defining a distribution (compatibility function) that associates the single observation vectors y_x^i , y_s^z for each of the possible outcomes, s_z , at node x_i . Here we use the binary attributes to construct the compatibility functions between neighbouring node states.

Assume that x_i and x_j are neighbours in the HMRF (are within the same clique of G_x). Similar to the unary attributes above, we define

$$A_{ji} = p(x_j|x_i) = \begin{pmatrix} C_2(y_x^{ij}, y_s^{11}) & \cdots & C_2(y_x^{ij}, y_s^{1S}) \\ \vdots & \ddots & \vdots \\ C_2(y_x^{ij}, y_s^{S1}) & \cdots & C_2(y_x^{ij}, y_s^{SS}) \end{pmatrix}$$

where each single element is given by $A_{ji;\beta\alpha} = p(x_j = s_\beta|x_i = s_\alpha) = C_2(y_x^{ij}, y_s^{\alpha\beta})$. We can also write A as a compatibility function, ψ

$$\psi_{ij} = \psi_{ij}(x_i, x_j) = p(x_j|x_i)$$

where each single element is given by $\psi_{ij\alpha\beta} = C_2(y_x^{ij}, y_s^{\alpha\beta})$ and, analogously to C_1 , C_2 is defined by

$$C_2(y_x^{ij}, y_s^{\alpha\beta}) = \mathcal{N}(y_x^{ij}; y_s^{\alpha\beta}, \text{cov})$$

where cov corresponds to the covariance matrix of the whole set of binary attributes of G_s .

So the complete HMRF model is defined by $\lambda = \{A, B\}$, having $2T$ nodes (pairs), T hidden nodes and T evidence or observation nodes. Each hidden node x_i (x_j) has S possible outcomes (which correspond to nodes of the graph G_s). For each pairwise combination of realizations of x_i and x_j , there is a scalar $p(x_j = s_\beta|x_i = s_\alpha)$ which measures the compatibility of this pairwise combination. For each pairwise combination of y_i (y_j) and a realization s_α (s_β) of x_i (x_j), there is also a scalar

$p(y_i|x_i = s_\alpha)(p(y_j|x_j = s_\beta))$ measuring the corresponding compatibility.

Given this general HMRF formulation for graph matching, the optimal solution is that of deriving a state vector $\underline{s}^* = (s^1, \dots, s^T)$, where $s^i \in G_s$ for each vertex x_i in G_x , such that the MAP criterion is satisfied, given the model λ and data Y ,

$$\underline{s}^* = \arg \max_{s_1, \dots, s_T} p(x_1 = s_1, \dots, x_T = s_T | \lambda, Y) \quad (1)$$

3. Matching algorithms

Since an exact solution to Eq. (1) is NP-hard for the fully connected MRF model, our approach to the solution may be either to find an approximated solution to the complete MRF or an exact solution to some feasibly sparse MRF. We will now present and compare a traditional solution to the problem, which is an example of the first approach, with the new solution that this paper presents, which is an instance of the second approach.

3.1. Probabilistic relaxation labeling

Probabilistic relaxation labeling (PRL) is a first-order recursive updating algorithm that has been used for graph matching in many areas of computer vision (Kittler and Hancock, 1989). Over the past decade it has been posed in Bayesian terms by Christmas et al. (1995) involving updating the belief (probability distribution for observation nodes within the HMRF) of a node's states as a function of its compatibility with its neighbouring states (the Markov property of the HMRF) and observations

$$\begin{aligned} b_i(X_i(t)) &= p(X_i(t)) \\ &= p(Y_i(t)|X_i(t)) \prod_{j \in \sigma_i} e_j^i(X_i(t)) \end{aligned} \quad (2)$$

where Y corresponds to the conditional observation pdf for X at position i . $e_j^i(X_i(t))$ is the evidence (message) passed from position j to position i about each state of the rv X at position i , at time t , and is given by

$$e_i^j(X_i(t+1)) = \sum_{X_j} [p(Y_j(t)|X_j(t)) \times p(X_j|X_i)e_i^j(X_j(t))] \quad (3)$$

In general, PRL does not require any special model structure. However, the clique and Markov properties of the HMRF restrict the updates in terms of local compatibilities within cliques. The procedure, itself, is heuristic and only an approximated MAP solution is obtained after the iterative process is stopped.

3.2. Single path dynamic programming

In contrast to PRL, single path dynamic programming (SPDP) is a simple model approximation (inexact model) that allows for exact, or optimal inference. That is, SPDP reduces the MRF to a single Markov chain defined by a selected path that transverses every vertex precisely once. With this assumption, the only binary features to be taken into account during the optimization process will be those between consecutive nodes in the chain. This is a considerable oversimplification, but it has the advantage that we can use dynamic programming to determine the (global) most likely labels for each node corresponding to the vertices of the matching graph. In PRL we can use all the binary information amongst pairs of nodes and still run the algorithm, but we cannot assure global optimality. Here, we do not use all the binary attribute information, but we assure global optimality for the approximated model.

We define the SPDP model as follows. Let x_1, \dots, x_T be an ordered sequence of nodes in G_x such that each node is visited exactly once. Again, T is the size of G_x and S is the size of G_s . Let the evidence potential $\phi(x_i = s_\alpha)$ denote the compatibility between the observation vector y_x^i and the observation vector y_s^α of the node s_α in G_s to which x_i is mapped. The Markovian potential $\psi(x_i = s_\alpha, x_j = s_\beta)$ denotes the compatibility of the joint assignment of x_i to s_α and x_j to s_β . The basic idea now consists in translating these terms into the well-known dynamic programming scheme for finding the best “state sequence” \underline{s}^* given the distribution of the observations (here translated as being the potentials, ϕ), and the distribution of

Markovian transitions (here seen as the potentials, ψ).

To solve this problem we define the recursive δ function as follows. First, we sample a path through the template graph. Then, for this path define

$$\delta_i(\alpha) = \max_{x_1, x_2, \dots, x_{i-1}} P(x_1, x_2, \dots, x_{i-1}, x_i = s_\alpha)$$

where we index the nodes in the path by variable i and the nodes to which they map by variable α . This can be solved by induction,

$$\delta_{i+1}(\beta) = \max_{\alpha} [\delta_i(\alpha)\psi(x_i = s_\alpha, x_{i+1} = s_\beta)] \times \phi(x_{i+1} = s_\beta)$$

where we must keep track of the maximizing arguments in each step, what we do via a variable ζ . The complete algorithm is a Viterbi-like algorithm (Rabiner, 1989), being different however in that the transition process is not stationary. The algorithm reads as follows

Initialization

For $1 \leq \alpha \leq S$,

$$\delta_1(\alpha) = \phi(x_1 = s_\alpha)$$

$$\zeta_1(\alpha) = 0$$

Recursion

For $2 \leq i \leq T$, $1 \leq \beta \leq S$,

$$\delta_i(\beta) = \max_{\alpha} [\delta_{i-1}(\alpha)\psi(x_i = s_\alpha, x_{i+1} = s_\beta)] \phi(x_{i+1} = s_\beta)$$

$$\zeta_i(\beta) = \arg \max_{\alpha} [\delta_{i-1}(\alpha)\psi(x_i = s_\alpha, x_{i+1} = s_\beta)]$$

Termination

$$p^* = \max_{\alpha} [\delta_T(\alpha)]$$

$$s_T^* = \arg \max_{\alpha} [\delta_T(\alpha)]$$

Reconstruction

For $i = T - 1, T - 2, \dots, 1$,

$$s_i^* = \zeta_{i+1}(s_{i+1}^*)$$

3.3. The junction tree framework

We now include richer cliques. The Junction Tree framework (Lauritzen, 1996) comprises a

set of algorithms that allows exact inference in arbitrary graphical models. However, the feasibility of its usage depends exactly on the structure of these models (specifically on the size of the maximal clique of the graph after it is triangulated). A Junction Tree is a tree where the nodes correspond to the maximal cliques of the underlying HMMF and the connectivity is such that the *junction tree property* is satisfied. This property states that all the “clique nodes” in a path from clique node i to clique node j must contain their intersection. This holds for triangulated graphs: graphs where every cycle of length >3 contains at least one chord (a chord in a cycle is an edge between non-consecutive nodes in that cycle; Jordan, in press).

The HUGIN algorithm (Lauritzen, 1996) is one of the possible techniques for optimal inference over Junction Trees. Fig. 1 shows the “tetragram” model (JT4) and its respective Junction Tree. Note that the nodes of the Junction Tree (circles in Fig. 1(b)) correspond precisely to the maximal cliques of the underlying tree (Fig. 1(a)). The “trigram” model (JT3) is analogous, but the hidden clique nodes are of size three, instead of four, and the connections between nodes x_i and x_{i+3} in Fig. 1(a) are not present. Note that the Junction Tree property is satisfied since every node in the path between any two given nodes contains the intersection of these two nodes. Note that there are *separators* between neighbour nodes, which we denote

with rectangular nodes. These separators include the set of singleton nodes that are common to both clique nodes and are introduced in order to apply the HUGIN algorithm.

The HUGIN algorithm essentially works in two steps: initialization and message-passing. During initialization, the potential of each separator is set to unity and the potential of each clique is introduced. In our particular case, the hidden clique potentials are products of the pairwise potentials that embodies the 4-size clique. So, the potential ψ_{ijkl} is actually given by $\psi_{ijkl} = \psi(x_i, x_j)\psi(x_i, x_k)\psi(x_i, x_l)$. The evidence potentials $\phi(y_i, x_i)$ are simply obtained as shown in the subsection 2.1. The second step is the message-passing scheme which involves a transfer of information between two nodes V and W . This operation is defined by the following update equations: $\phi_S^* = \max_{V \setminus S} \{\psi_V\}$ and $\psi_W^* = \frac{\phi_S^*}{\phi_S} \psi_W$. These potential update rules must respect the following protocol: a node V can only send a message to a node W when it has already received messages from all its other neighbours. If this protocol is respected and the equations are applied until all clique nodes have been updated, the algorithm assures that the resulting potential in each node and separator is equal to the maximum probability of the set of enclosed singleton nodes (Jordan, in press). In our particular case, we need the maximum probability for each singleton, what can be obtained by

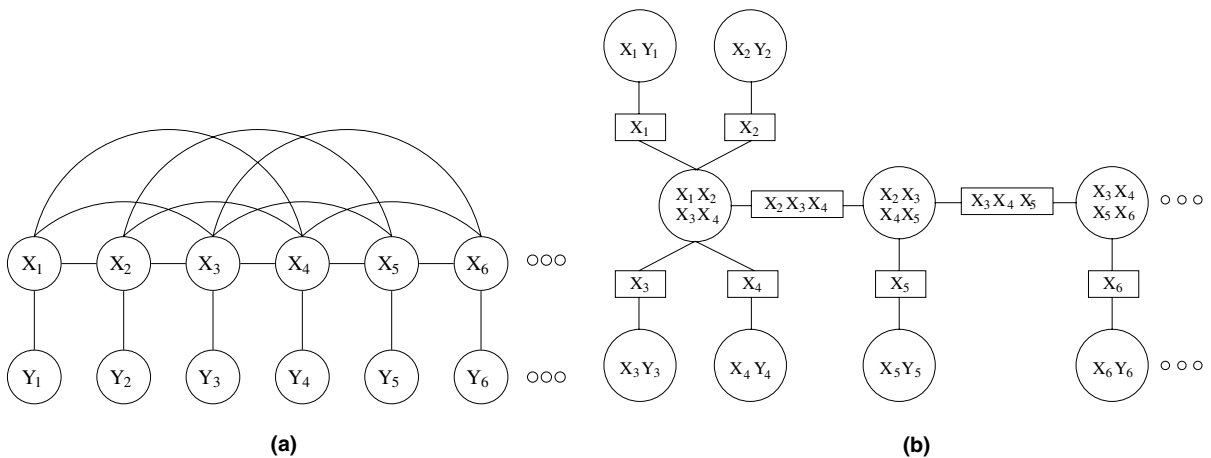


Fig. 1. (a) A tetragram model and (b) a corresponding Junction Tree.

observing the final potential of the separators that contain a single node (see Fig. 1(b)).

4. Experiments and results

We have performed extensive experiments on two problems: attributed (feature) graph matching and point set matching. The former consisted of sets of oriented line segments as shown in Fig. 2, the latter only using positional information—both allowing us to objectively control complexity and noise perturbations of templates and scenes.

In the first problem we compared the performances of PRL, SPDP, JT3 and JT4. The underlying MRF for PRL was set as a fully connected graph (the “complete” model) and the experiments consisted of matching the straight line segments (SLSs). The “template” consisted of a series of SLSs generated according to normal distributions of length, angle and position. The “scene”, or “signal plus noise” images contained the template *and* a set of noisy SLSs also generated from controlled normal distributions for length, angle and position. The unary feature used was the length, and the binary features used were (i) relative angle, (ii) relative distance of centroids, (iii) relative maximal distance and (iv) relative minimal distance between extremal points of the SLSs.

The first experiment consisted of studying the stability of the results while increasingly resetting

the amount of noisy SLSs, for a fixed amount of SLSs in the template. For each setting, we took one thousand runs, and the average performance for each experiment is presented in Fig. 3(a). The performance of PRL is severely affected, whereas those of the proposed models remain fairly robust. This result agrees with previous results in the literature, which have shown that PRL is sensitive to the complexities of more naturally occurring attributed graph matching problems (Gold and Rangarajan, 1996).

In the second experiment, we held constant the amount of SLSs in the template and the scene (10 and 20, respectively), but perturbed each vector graph contained in the template instance in the scene. Each extremal point was shifted isotropically by a random number drawn from a normal distribution with zero mean and varying standard deviation corresponding to the x axis in Fig. 3(b). This simulates noisy situations, what is almost always the case in inexact attributed graph matching, when the feature extraction process introduces noise in the measured attributes making the matching algorithm sensitive to the degree of distortion. We can observe in Fig. 3(b) that JT4 and JT3 performed better than PRL under noise augmentation.

A similar procedure was used for the point set matching experiments, except that we only compared PRL with JT4. In this problem, the only binary feature in the MRF is the pairwise distance

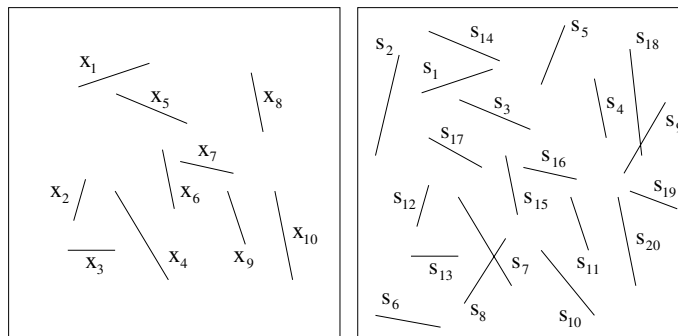


Fig. 2. Shows images used for comparing matching algorithms. Here the left image shows a “signal” which is embedded in the right “scene: signal-plus-noise” image. Both signal and noise component positions, orientation and lengths are all normally distributed and systematically controlled to test the robustness of the matching task. A perfect match would be (x_1, s_1) , (x_2, s_{12}) , (x_3, s_{13}) , (x_4, s_7) , (x_5, s_3) , (x_6, s_{15}) , (x_7, s_{16}) , (x_8, s_4) , (x_9, s_{11}) and (x_{10}, s_{20}) .

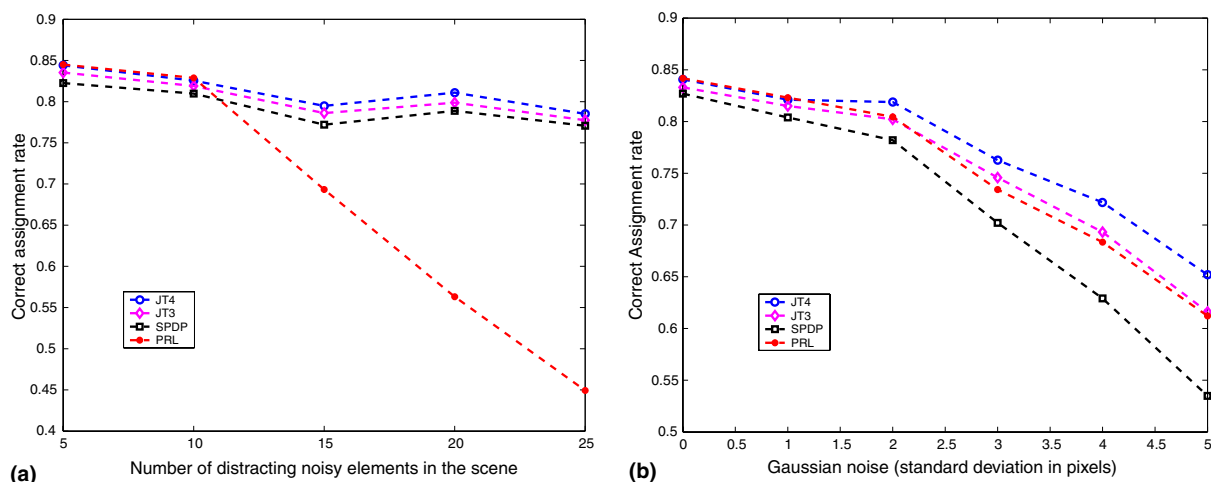


Fig. 3. Performances of the Junction Tree tetragram model (JT4), Junction Tree trigram model (JT3), single path dynamic programming (SPDP) and Probabilistic relaxation labeling (PRL). (a) Performance variation according to the task complexity (extrinsic noise); (b) Performance variation according to degree of perturbation in the query template (intrinsic noise).

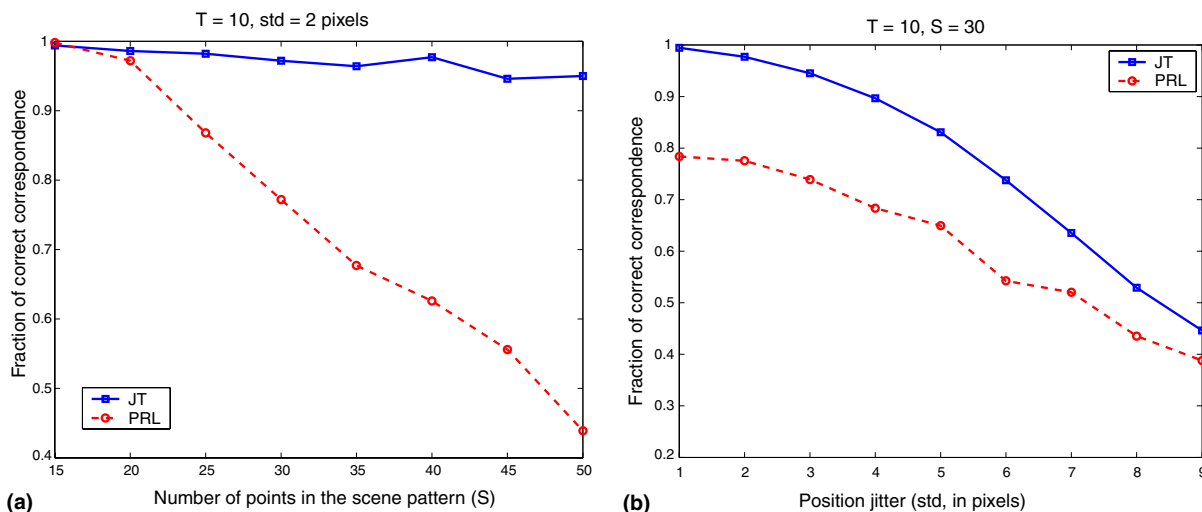


Fig. 4. (a) Performances of JT and PRL when the size of the template pattern is increased; std (noise jitter) = 2 pixels. (b) Performances of JT and PRL when the noise jitter (std) in the domain pattern is increased; $T = 10$, $S = 30$.

between the points (there is no unary feature, so the observation component is not present). Fig. 4 shows the results for the point set matching problem. In Fig. 4(a), we evaluated the stability of the algorithms by increasing the number of points in the scene pattern. JT4 clearly presents better stability, resembling the results obtained for the line

segments matching problem. In Fig. 4(b), the sizes of both patterns are fixed and gaussian noise is added to the scene pattern. We can also observe that JT4 has better performance, although we must expect that for extremely large perturbations the performances will be similar due to impossibility of performing significantly better than choice.

The computational complexity of JT and PRL are $O(TS^c)$ and $O(T^2S^3)$, respectively, being c the size of the maximal clique in the model (e.g., $c = 3$ for the trigram model and $c = 4$ for the tetragram model). As a result, JT3 is less expensive than PRL, whereas JT is more expensive if $S > T$. However, it is worth noting that, whereas PRL is an iterative procedure and does local optimization, JT is a two-pass algorithm that performs global optimization.

5. Discussion and conclusion

To this date exact and inexact matching have not been analyzed in terms of HMRFs, where the key idea involves posing the matching problem in terms of deriving the optimal labeling of one graph's vertices with respect to the labels of the matching graph where each vertex is a random variable in an embedding MRF. Understanding graph matching in this way allows us to use powerful exact inference methods from Graphical Models for attributed graph matching in general. By exploiting the Markovian properties of simple triangulated graph structures which have a fixed maximal clique size, it is possible to perform exact probabilistic inference in polynomial time. The proposed technique is much more robust than standard probabilistic relaxation labelling to varying point set sizes when under noise. The technique is also robust with respect to the augmentation of the noise level, when it clearly outperforms standard probabilistic relaxation labelling.

These results demonstrate the trade-off between the exactness of the data model and the optimality of the algorithm. At one end of the modeling spectrum, PRL used an "exact" model (the fully connected graph model) and an inexact inference algorithm. At the other end, SPDP used an impoverished data model (single Hamiltonian path) and an exact (optimal) inference algorithm. Both these cases led to inferior performance to the JT algorithms which used an optimal algorithm but with richer data models. JT4 was richer than JT3 and

this was reflected in performance: JT4 performs best over all algorithms.

References

- Bunke, H., Shearer, K., 1998. A graph edit distance metric based on the maximum common subgraph. *Pattern Recognition Lett.* 19, 255–259.
- Caelli, T., Cheng, L., Feng, Q., 2003. A Bayesian approach to image understanding: From images to virtual forests. In: 16th Internat. Conf. on Vision Interface. Springer: ISSN 0843-803X.
- Caelli, T., Kosinov, S., 2004. An eigenspace projection clustering method for inexact graph matching. *IEEE Trans. PAMI* 26 (4), 515–519.
- Cheng, H., Bouman, C.A., 2001. Multiscale Bayesian segmentation using a trainable context model. *IEEE Trans. Image Process.* 10 (4), 511–525.
- Christmas, W.J., Kittler, J., Petrou, M., 1995. Structural matching in computer vision using probabilistic relaxation. *IEEE Trans. PAMI* 17 (8), 749–764.
- Fu, K.S., 1983. A step towards unification of syntactic and statistical pattern recognition. *IEEE Trans. PAMI* 5 (2), 200–205.
- Gold, S., Rangarajan, A., 1996. A graduated assignment algorithm for graph matching. *IEEE Trans. PAMI* 18 (4), 377–388.
- Jordan, M.I., in press. *An Introduction to Probabilistic Graphical Models*. MIT Press, Cambridge, MA.
- Kittler, J., Hancock, E., 1989. Combining evidence in probabilistic relaxation. *PRAI* 3, 29–51.
- Lauritzen, S.L., 1996. *Graphical Models*. Oxford University Press, New York, NY.
- Pelillo, M., Siddiqi, K., Zucker, S., 1999. Matching hierarchical structures using association graphs. *IEEE Trans. Pattern Anal. Machine Intell.* 21, 1105–1120.
- Rabiner, L.R., 1989. A tutorial on hidden Markov models and selected applications in speech recognition. *Proc. IEEE* 77 (2), 257–286.
- Shapiro, L., Brady, J., 1992. Feature-based correspondence—an eigenvector approach. *Image Vision Comput.* 10, 283–288.
- Tsai, W.H., Fu, K.S., 1979. Error-correcting isomorphism of attributed relational graphs for pattern analysis. *IEEE Trans. Systems Man Cybernet.* 9 (2), 757–768.
- van Wyk, M.A., Durrani, T.S., van Wyk, B.J., 2002. A rkhs interpolator-based graph matching algorithm. *IEEE Trans PAMI* 24 (7), 988–995.
- Zhang, Y., Brady, M., Smith, S., 2001. Segmentation of brain mr images through a hidden Markov random field model and the expectation maximization algorithm. *IEEE Trans Medical Imaging* 20 (1), 45–57.