Graphical Models for Graph Matching

Tibério S. Caetano[†‡],Terry Caelli[†] and Dante A. C. Barone[‡]

†Department of Computing Science
University of Alberta
Edmonton, AB, Canada, T6G 2E8

‡Instituto de Informática
Universidade Federal do Rio Grande do Sul
Porto Alegre, RS, Brazil, CP 15064

Abstract

This paper explores a formulation for attributed graph matching as an inference problem over a hidden Markov Random Field. We approximate the fully connected model with simpler models in which optimal inference is feasible, and contrast them to the well-known probabilistic relaxation method, which can operate over the complete model but does not assure global optimality. The approach is well suited for applications in which there is redundancy in the binary attributes of the graph, such as in the matching of straight line segments. Results demonstrate that, in this application, the proposed models have superior robustness over probabilistic relaxation under additive noise conditions.

1. Introduction

The general problem of Attributed Graph Matching (AGM) has a long history, dating back to the pioneering work of Fu [3] on Attributed Relational Graphs (ARGs). Inexact graph matching and multi-subgraph matching have been extensively studied by Bunke [1] and coworkers by extending methods to include distance measures such as edit distance. Recently, the vision and pattern recognition community has explored two new classes of solutions. The first is based on deterministic linear least squares [14] and graph eigenspectra [12, 7]. The second, of interest in this paper, are those based on probabilistic optimization models, including well-known Probabilistic Relaxation Labeling (PRL), which has been used for graph matching over the past decade [10, 2]. In particular, Hancock and associates [5] have made significant contributions to a probabilistic framework for graph matching using versions of Probabilistic Relaxation.

This work aims at further formalizing the probabilistic optimization approach to graph matching under the theory of Probabilistic Graphical Models [8], providing a formal alternative to PRL, which is, essentially, a heuristic method [2]. In this new theoretical context, we propose models for attributed graph matching in terms of an underlying hidden Markov Random Field (HMRF) image feature graph model. The basic idea behind our proposal is that in most real AGM applications there is a significant correlation of binary information between different pairs of nodes, and conditional independence assumptions may be postulated in order to simplify the original fully connected ARG while still retaining good performance. This naturally gives rise to a Markov Random Field formulation, where only "local" interactions are used.

This simplification enables us to use models in which *exact* probabilistic inference is feasible, in contrast to most techniques for AGM, which present approximate solutions to the original problem involving fully connected graphs. Specifically, we introduce two sets of HMRF models for solving the AGM problem. The first, Single Path Dynamic Programming (SPDP), consists in approximating the fully connected graph with a single chain which traverses each vertex exactly once. This model allows us to use Dynamic Programming in order to find the optimal match between the graphs. In the second type of model, we improve over this simple chain model by proposing HMRFs with maximal cliques of size 3 and 4, what retains the feasibility of the model and enables optimal inference via Junction Tree methods [8].

The paper is organized as follows. In section 2, our formulation of AGM as a probabilistic optimization problem defined over a HMRF is presented. Section 3 describes the

Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'04)
1063-6919/04 $20.00 © 2004 **IEEE**

specific problem domain in which the proposed models will be applied. In section 4, we propose HMRF models for AGM. Section 5 is dedicated to comparing PRL to the models proposed in a set of controlled experiments. In section 6 we present results and discussion and, finally, in section 7 we draw conclusions.

2. Graph Matching Models as HMRFs

In this section we formulate AGM as an optimization problem defined over a HMRF. The two different kinds of proposed models will then be variations where specific structure (connectivity) and algorithms are adopted in this HMRF representation.

Assume we have two ARGs, G_s and G_x, representing the scene and the template/query, respectively. The scene consists of the template plus a set of "background" or "noisy" nodes. We define a HMRF on the template graph G_x. A single node in the graph G_x is defined by x_i, and in the graph G_s, s_α. Each node in each graph has a unary attribute vector: y_x^i in G_x and y_s^α in G_s, where $1 \leq i \leq T$ and $1 \leq \alpha \leq S$, being S and T 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 $y_s^{\alpha\beta}$ and y_x^{ij}, representing the relational attribute vectors between vertices α, β in G_s and between vertices i, j in G_x.

The subgraph isomorphism problem involves assigning to each x_i a unique s_α - assuming that there is one "signal" embedded in the "scene". In this paper we formulate this in terms of optimal inference over HMRFs, as follows. We consider each node x_i in G_x as a discrete random variable (r.v.) which can assume any of S possible values, s_α, corresponding to the nodes of G_s.

2.1. 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^i whose values, we will see, are dependent on the "states" of x_i corresponding to the vertices of G_s. That is, y_x^i is dependent only on its vertex x_i, as:

$$B_i = p(y_x^i|x_i) = [\mathcal{S}(y_x^i, y_s^1), \cdots, \mathcal{S}(y_x^i, y_s^S)]^t$$

where each single element is given by

$$B_{i\alpha} = p(y_x^i|x_i = s_\alpha) = \mathcal{S}(y_x^i, y_s^\alpha)$$

and where y_s^α is the unary attribute associated to the discrete value s_α of r.v. x_i. The function $\mathcal{S}$ is a similarity function: it is zero for totally incompatible features and 1 for totally compatible features. Here we have used a multivariate Gaussian for this purpose and the similarity functions are given by

$$\mathcal{S}(a, b) = \mathcal{N}_a(b, cov)$$

where *cov* is a diagonal covariance matrix (assuming attribute independence) such that each eigenvalue is the sample variance of the complete set of unary attributes in G_s. This merely weights the different attributes so that they can be gathered into a single representational measure.

Consequently the conditional density function, $p(y_x^i|x_i)$, is a compatibility function $\phi_i(y_x^i, x_i)$ that defines, for each possible outcome of x_i, the compatibility between y_x^i and y_s^α. That is:

$$\phi_i(y_x^i, x_i) = B_i$$

where subscript i in ϕ_i denotes that this vector is dependent solely on i. In the same way we write the single compatibility coefficient $\phi_{i\alpha}$ as

$$\phi_{i\alpha} = \mathcal{S}(y_x^i, y_s^\alpha).$$

2.2. Markov Component

Accordingly, 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^α for each of the possible values s_α, at node x_i. Here we use the binary attributes to construct the compatibility functions between states of neighboring nodes.

Assume that x_i and x_j are neighbors in the HMRF (being connected in G_x). Similarly to the unary attributes above, we define:

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

where each single element is given by

$$A_{ji;\beta\alpha} = p(x_j = s_\beta|x_i = s_\alpha) = \mathcal{S}(y_x^{ij}, y_s^{\alpha\beta}).$$

We can also write A as a compatibility function, ψ:

$$\psi_{ij}(x_i, x_j) = A_{ji}$$

where each single element is given by

$$\psi_{ij\alpha\beta} = \mathcal{S}(y_x^{ij}, y_x^{\alpha\beta})$$

and $\mathcal{S}$ is defined by

$$\mathcal{S}(a, b) = \mathcal{N}_a(b, cov)$$

where *cov* is a diagonal covariance matrix (again, assuming attribute independence) with eigenvalues equal to the sample variance of the complete set of binary attributes of G_s.

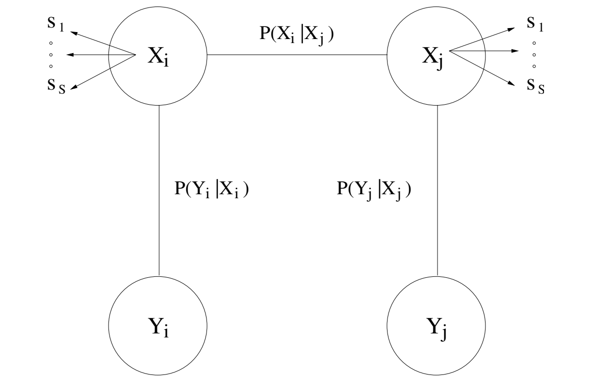


Figure 1. An instance of the HMRF model. Observation conditions (vertical lines) are defined by the dependencies of observations (node attributes) on node states (labels) and the Markov condition (horizontal line) is defined by the interdependencies of node labels corresponding to the nodes of the scene graph. Each hidden node x_i can assume S different states.

The complete HMRF model is defined by

$$\lambda = \{A, B\}$$

having 2T nodes, being T hidden nodes and T evidence or observation nodes.

Figure 1 shows a graphical representation of the model and parameters. 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 and a realization s_α of x_i, there is also a scalar $p(y_i | x_i = x_\alpha)$ measuring the corresponding compatibilities.

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

$$\underline{s}^* = arg \max_{s_\alpha, \cdots, s_\zeta} p(x_1 = s_\alpha, \cdots, x_T = s_\zeta | \lambda, Y) \qquad (1)$$

where Y denotes the whole set of attributes in both graphs.

3. The Problem Domain

In order to access quantitatively the performance of the proposed models, we have prepared a simulated controlled environment. We have defined a set of vector-graph [14, 9] patterns consisting of spatial distributions of oriented straight line segments (SLSs). In this case the position, orientation and length distributions of these lines are defined by normally distributed r.v.'s and they can also be embeded in other sets of lines ("background"), whose parameter values are also normally distributed as shown in Figure 2. The purpose then is to find a mapping from the "signal" vector-graph to the "signal-plus-noise" vector graph.

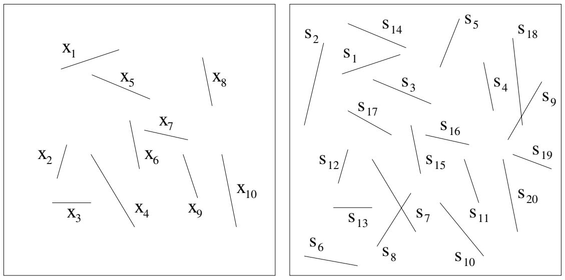


Figure 2. Shows the type of images used for comparing matching algorithms. Here the left image shows a vector-graph "signal" which is embedded in the right "scene: signal-plus-noise" image. Both signal and noise component positions, orientation and lengths are all normally distributed and systematically 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}).

Note that there is a natural "redundancy" in the set of binary attributes in this type of problem domain in 2D. For example, consider a very natural choice for a binary feature: the relative angle of two SLSs. In this case, if we know the feature for pairs (A,B) and (B,C), then the feature for pair (A,C) is uniquely determined, and we do not need to include it explicitly during the optimization procedure. In another example, assume the binary feature "relative distance of centroids". If we know this feature for each pair of nodes (A,B,C,D) and for each pair of nodes (B,C,D,E), then the feature between (A,E) is uniquely determined.

The existence of this kind of redundancy in binary attributes suggest that several edges in the underlying ARG may actually be conveying no new information whatsoever. In other words, given that we know a subset of binary relations, there are other subsets of binary relations that add no new information about the structure. It is exactly this observation that motivated us to propose the models which appear in the following section.

4. Proposed Models

Since a complete solution to Equation (1) is N-P hard, various approximations to the solution have been developed. In this section we present our approach, which consists in approximating the *problem* instead of the solution, by identifying approximate models for the original ARG and finding the exact solution for these models. We introduce two types of models that simplify the fully connected ARG within a Markov structure. Along with each model we also present algorithms that assure a global optimum for the probabilistic optimization procedure.

4.1. Single Path Dynamic Programming

Single Path Dynamic Programming (SPDP) is a simple model approximation in so far as we assume that the HMRF reduces to a single Markov chain defined by a selected path that transverses each 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 most likely labels for each node corresponding to the vertices of the matching graph. In particular, we find it appropriate to start with a brief description of Probabilistic Relaxation Labeling and then point the key differences from SPDP. Standard PRL equations are as follows [2]:

$$b_i(X_i(t)) = p(X_i(t)) = p(Y_i(t)|X_i(t)) \prod_{j \in \sigma_i} e_i^j(X_i(t)) \tag{2}$$

where $e_i^j(X_i)$ is the evidence (message) passed from position j to position i about each state of the r.v. X at position i - at time t, as given below:

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

where Y corresponds to the conditional observation pdf for X at position i. Note that in PRL the neighborhood for each node can be the entire set of remaining nodes, thus we can use all the binary information available from the feature extraction process. However, PRL is known to converge only to a local optimum [2]. The model that we present here, SPDP, contrasts with PRL in that we do not use all the information of binary attributes, but we assure global optimality for the approximated model.

We can define the model as follows. Let $x_1, .., x_T$ be an ordered sequence of nodes in G_x such that each node is visited exactly once. The definitions of Section 2 hold here, thus T and S correspond to the number of vertices in G_x and G_s, respectively. Also lets recall the notation for the clique potentials. The evidence potential $\phi(x_i = s_\alpha)$ - where we dropped the dependency on y_x^i due to the fact that it is a constant - denotes 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 observations (here translated as being the unary potentials, ϕ) and the distribution of Markovian transitions (here seen as the binary potentials, ψ).

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

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

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, via a variable ξ. The complete algorithm is a Viterbi-like algorithm [11], and read as follows:

Initialization

$$\text{For } 1 \le \alpha \le S, \\ \delta_1(\alpha) = \phi(x_1 = s_\alpha) \\ \xi_1(\alpha) = 0$$

Recursion

$$\text{For } 2 \le i \le T, 1 \le \beta \le S, \\ \delta_i(\beta) = \max_\alpha [\delta_{i-1}(\alpha)\psi(x_i = s_\alpha, x_{i+1} = s_\beta)] \\ \times \phi(x_{i+1} = s_\beta) \\ \xi_i(\beta) = arg \max_\alpha [\delta_{i-1}(\alpha)\psi(x_i = s_\alpha, x_{i+1} = s_\beta)]$$

Termination

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

Reconstruction

$$\text{For } i = T-1, T-2, ..., 1, \\ s_i^* = \xi_{i+1}(s_{i+1}^*)$$

This result is approximately independent of the path chosen but variations can be due to the rounding errors of different lists of MAP values. However, in principle, these results should be similar as the algorithm is based on a non-stationary dynamic programming model where all lists are based on local unary and binary computations. For this reason the algorithm is a generalization of the classical Viterbi algorithm that assumes a stationary hidden Markov model to compute MAP lists.

4.2. Junction Tree Models

The Junction Tree framework [8] comprises a set of algorithms that allows exact inference in arbitrary graphical models, though the feasibility of its usage depends on the complexity of these models (specifically, on the size of the maximal clique of the graph after its triangulation).

A Junction Tree is a graph where the nodes correspond to the maximal cliques (we call here "clique nodes") of the underlying HMRF and such that the *junction tree property* is satisfied. This property merely states that all the clique nodes in a path between two given clique nodes must contain their intersection (the singleton nodes of the original HMRF that are in both clique nodes). It is known that the condition for the existence of a Junction Tree is that the graph must be triangulated [6].

Here, we propose two HMRF models and apply the Hugin algorithm [8], an instance of the Junction Tree framework, for accomplishing exact inference.

Figure 3 shows the "tetragram" model (JT4) and its respective Junction Tree. The "trigram" model (JT3) is analogous, but the hidden node cliques are of size three, instead of four, and the connections between nodes x_i and x_{i+3} in Figure 3(a) are not present. Note, in Figure 3(b), that the Junction Tree property is satisfied, once all paths from any node to any other contains the intersection of these nodes. Note also that there are *separators* between neighbor nodes, which we denote with rectangles. 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.

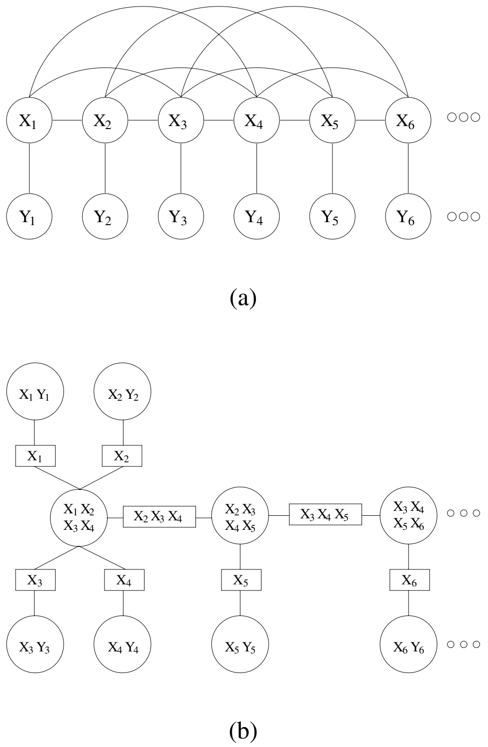


Figure 3. (a) A tetragram model and (b) a corresponding Junction Tree.

The Hugin algorithm essentially works in two steps: initialization and message-passing. During the 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 $\psi(x_i, x_j, x_k, x_l)$ is actually given by $\psi(x_i, x_j)\psi(x_i, x_k)\psi(x_i, x_l)\psi(x_j, x_k)\ \psi(x_j, x_l)\psi(x_k, x_l)$, but each pairwise potential must be included just once, so, actually, this product involves these six factors just for a single clique node, and it is not difficult to see that there are just three factors for the others. The evidence potentials $\phi(y_i, x_i)$ are simply obtained as shown in section 2. The second step is the message-passing scheme, which involves a transfer of information between two nodes V and W [6]. This operation is defined by the following equations:

$$\Phi_S^* = \max_{V \setminus S} \Psi_V \tag{4}$$

$$\Psi_W^* = \frac{\Phi_S^*}{\Phi_S} \Psi_W \tag{5}$$

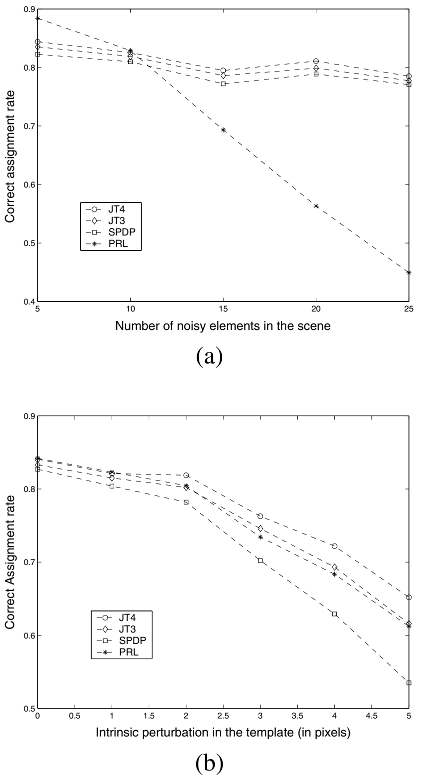
where we used standard notation for the current and updated (*) versions of the separator potentials (Φ) and the clique potentials (Ψ). The first equation is a maximization over all sub-configurations in Ψ_V that do not involve the singleton nodes which are common to Φ_S and Ψ_V. The second is simply a normalization step necessary to keep Ψ_W consistent with the updated version of Φ_S. The above 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 neighbors. 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 a posteriori probability distribution of the set of enclosed singleton nodes [6]. In our particular case, we need the maximum probability for each singleton, that can be obtained by observing the index that maximizes the final potential of the separators that contain a single node (see separators X_i in Figure 3(b)).

5. Experiments

We have compared the performances of PRL, SPDP, JT3 and JT4 in a set of experiments. The underlying HMRF for PRL was set as the fully connected graph, so that the algorithm could take advantage of all the binary information available. The experiments consisted of matching straight line segment (SLS) vector graphs. The “template” vector graph consists of a series of SLSs generated according to normal distributions of length, angle and position. The “scene”, or “signal plus noise” vector graph contains the template vector graph *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 [13]. These features are translational and rotational invariant.

In order to access the performances of the four methods, we accomplished two sets of experiments. The first consists in studying the stability of the results when increasing the amount of noisy SLSs. We have used a template of 10 SLSs and evaluated the correct assignment performance for a scene with 5, 10, 15, 20 and 25 noisy SLSs, thus running a set of 5 experiments, (10,15), (10,20), (10,25), (10,30) and (10,35), where (T,S) denotes an experiment with T SLSs in the template and S SLSs in the scene. Figure 4(a) shows the performance for 1000 trials of each of these experiments.

The second experiment was carried out aimed at addressing the performance of the four methods when the template pattern, itself, is perturbed in some way. We kept constant the amount of SLSs in the template and the scene (10 in the template and 20 in the scene), but perturbed each SLS contained in the template instance in the scene. Each of their extremal points was shifted, both in x and y coordinates, by a random number drawn from a normal distribution with zero mean and varying standard deviation - given, in pixels, in the x axis in Figure 4(b).



(a)

(b)

Figure 4. 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 (number of noisy SLSs or extrinsic noise); (b) Performance variation according to degree of perturbation in the query template (standard deviation, in pixels, or intrinsic noise).

For the purpose of evaluating the algorithms in a real situation, we applied them to the problem of matching aerial images of the same region taken from different views. We present here a single example, so our aim is not to claim generality, but just to illustrate how the system works in practice. In the particular example to be shown, the amount of SLSs in both the template and scene graphs is similar, what is common in applications such as stereo matching. However, this may not be the case in general, when frequently the template is much smaller than the scene. Figure 5(a) shows two different aerial views of the same region, where some landmark lines summarizing its content were extracted using a set of Gabor filter banks in multiple scales and orientations (Figure 5(b)).

We applied PRL, SPDP, JT3 and JT4 to this problem. We defined the HMRF over the left image in Figure 5(b), so that the states assumed by the nodes (SLSs of this image) corre-

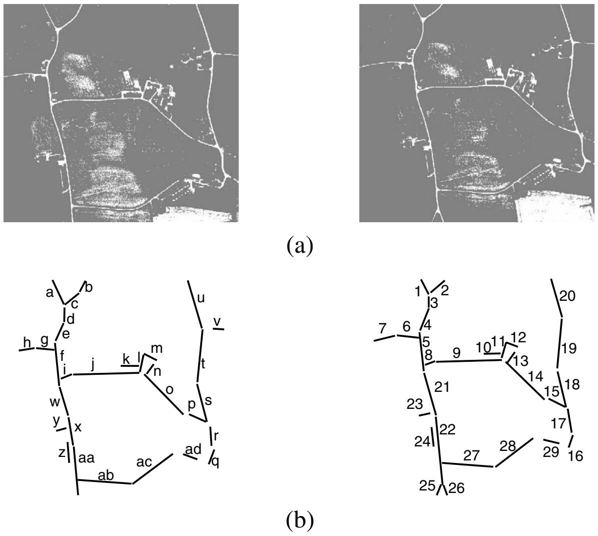


Figure 5. (a) Left: first aerial view. Right: second aerial view (b) Left: extracted lines of first aerial view. Right: extracted lines of second aerial view.

spond to the SLSs in the right image in the figure. We obtained the following mapping results:

- PRL:(a,3) (b,2) (c,2) (d,4) (e,4) (f,5) (g,6) (h,7) (i,4) (j,27) (k,10) (l,11) (m,10) (n,13) (o,15) (p,15) (q,16) (r,17) (s,18) (t,18) (u,20) (v,10) (w,21) (x,22) (y,23) (z,24) (aa,21) (ab,28) (ac,28) (ad,17).
- SPDP: (a,1) (b,2) (c,2) (d,4) (e,4) (f,5) (g,6) (h,7) (i,6) (j,9) (k,10) (l,13) (m,12) (n,13) (o,14) (p,14) (q,16) (r,16) (s,18) (t,19) (u,19) (v,17) (w,21) (x,22) (y,23) (z,24) (aa,22) (ab,27) (ac,28) (ad,17).
- JT3: (a,1) (b,2) (c,2) (d,4) (e,4) (f,5) (g,6) (h,7) (i,8) (j,9) (k,10) (l,10) (m,12) (n,13) (o,14) (p,15) (q,16) (r,18) (s,18) (t,19) (u,20) (v,13) (w,21) (x,22) (y,27) (z,24) (aa,22) (ab,27) (ac,28) (ad,15).
- JT4: (a,1) (b,2) (c,2) (d,4) (e,4) (f,5) (g,6) (h,6) (i,8) (j,9) (k,10) (l,13) (m,12) (n,13) (o,14) (p,15) (q,16) (r,16) (s,18) (t,19) (u,20) (v,10) (w,21) (x,22) (y,23) (z,24) (aa,22) (ab,27) (ac,28) (ad,15).

By inspection we can verify that in this particular case PRL was outperformed by the other three models, and JT3 and JT4 exhibited the best performance, though both made some mistakes. Note that the feature extraction process generated a significant amount of noise, since for example we can see that there are SLSs in one image that do not appear in the other, or even that some SLSs in one image have been detected as being two SLSs in the other.

6. Results and Discussion

Figure 4(a) shows that when the task complexity increases (when the number of noisy patterns added to the scene image grows) the performance of PRL is severely affected, while that of the proposed models remain fairly robust. This is an important result, since in real applications, such as road detection in aerial images, the scene may be full of patterns, frequently in amounts orders of magnitude greater than those of the template itself. This result agrees with previous results in the literature, which have shown that PRL performance is sensitive to complex attributed graph matching problems [4].

Similarly, at least one of our models (JT4) presents better performance when compared to PRL for addition of intrinsic noise in the template pattern embedded in the scene, as Figure 4(b) shows. This 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. This fact leads us to conclude that robustness with respect to intrinsic noise is very important in matching applications which reinforces the viability of the models proposed in this paper.

As a final remark, it is important to compare the computational complexity of the different techniques. PRL, SPDP, JT3 and JT4 are, respectively, $O(S^3T^2)$, $O(S^2T)$, $O(S^3T)$ and $O(S^4T)$. Here we can see that JT3, which outperforms PRL in robustness to extrinsic noise and has practically the same overall robustness to intrinsic noise, is cheaper. Also JT4, which clearly outperforms PRL in both situations, is S/T times more expensive than PRL, which in the example in Figure 5 corresponds to a factor of practically 1, like in stereo matching. In other applications, where $S >> T$, we may have to pay additional computational power in order to acquire more robustness by using JT4.

These computational complexity results, together with the evidence obtained from the reported experiments, encourage us to believe that the ideas presented in this paper may suggest a feasible foundation for providing more principled treatments, as well as more accurate techniques, for probabilistic optimization approaches to attributed graph matching.

7. Conclusions

We have formulated attributed graph matching as an optimization problem in a HMRF. We have also examined different types of HMRF models for the attributed graph matching problem, and compared their performance with the standard technique of Probabilistic Relaxation Labeling. The fundamental idea behind our proposal is that accomplishing exact inference in approximate models may be a better alternative to performing approximate optimization in complete models when there is redundancy of binary attributes. Indeed, the PRL approach can take advantage of using all the binary attributes available, in the sense that the graph underling the process can be the fully connected graph. However, due to its local optimization characteristics, the results, even with all the binary information made explicit, have shown to be poorer than those obtained with some of the models proposed, which do not use the entire explicit information available, but assure a solution which is a global optimum for the given approximated model.

Acknowledgements

We are grateful to NSERC/Canada and CAPES/Brazil for financial support.

References

[1] H. Bunke. Recent advances in structural pattern recognition with application to visual form analysis. *IWVF4, LNCS*, 2059:11–23, 2001.

[2] W. J. Christmas, J. Kittler, and M. Petrou. Structural matching in computer vision using probabilistic relaxation. *IEEE Trans. PAMI*, 17(8):749–764, 1994.

[3] K. S. Fu. A step towards unification of syntatic and statistical pattern recognition. *IEEE Trans. PAMI*, 5(2):200–205, 1983.

[4] S. Gold and A. Rangarajan. A graduated assignment algorithm for graph matching. *IEEE Trans. PAMI*, 18(4):377–388, 1996.

[5] E. Hancock and R. C. Wilson. Graph-based methods for vision: A yorkist manifesto. *SSPR & SPR 2002, LNCS*, 2396:31–46, 2002.

[6] M. I. Jordan. *An Introduction to Probabilistic Graphical Models*. In preparation.

[7] S. Kosinov and T. Caelli. Inexact multisubgraph matching using graph eigenspace and clustering models. *SSPR & SPR 2002, LNCS*, 2396:133–142, 2002.

[8] S. L. Lauritzen. *Graphical Models*. Oxford University Press, New York, NY, 1996.

[9] S. Z. Li. Matching: Invariant to translations, rotations and scale changes. *Pattern Recognition*, 25(6):583–594, 1992.

[10] B. Luo and E. Hancock. Structural graph matching using the em algorithm and singular value decomposition. *IEEE Trans. PAMI*, 23(10):1120–1136, October 2001.

[11] L. R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[12] L. Shapiro and J. Brady. Feature-based correspondence - an eigenvector approach. *Image and Vision Computing*, 10:268–281, 1992.

[13] P. N. Suganthan. Structural pattern recognition using genetic algorithms. *Pattern Recognition*, 35:1883–1893, 2002.

[14] M. A. van Wyk, T. S. Durrani, and B. J. van Wyk. A rkhs interpolator-based graph matching algorithm. *IEEE:PAMI*, 24(7):988–995, 2002.