

Stochastic Attributed Relational Graph Matching for Image Near-Duplicate Detection

DVMM Technical Report #206-2004-6

Dong-Qing Zhang Shih-Fu Chang

Department of Electrical Engineering

Columbia University

New York City, NY 10027

Abstract

Attributed Relational Graph (ARG) is a useful model for representing many real-world relational patterns. Computing the similarity of ARGs is a fundamental problem for ARG based modelling. This report presents a novel stochastic framework for computing the similarity of ARGs, which defines the ARG similarity as the likelihood ratio of the stochastic process that transforms one ARG to the other. We show that the transformation likelihood can be factorized and approximately calculated using variational approximation and Loopy Belief Propagation. Furthermore, we show that the similarity measure can be learned from training data using a variational E-M process in an unsupervised manner by using graph-level annotations. We apply the technique to visual scene matching and establish a part-based similarity measure for detecting Near-Duplicate Images in video database.

1 Introduction

Attributed Relational Graph (ARG) generalizes the ordinary graph by attaching discrete or continuous feature vectors to its vertexes and edges. Many real-world entities can be represented as ARG. For example, visual scenes can be modelled as the composition of the objects with specific spatial/attribute relationship. Molecular compounds can be represented as ARG with atoms as vertexes and bonds as edges. A web site can be represented as an ARG with web pages as vertexes and URL links as edges. A fundamental problem of the ARG based modelling is to compute the similarity of two ARGs, which is important for retrieval and mining applications.

Research on ARG matching can be dated back to the seminal work of Barrow [3]. In the past decades, there have been much research work on defining ARG similarities and the development of the computational schemes. The major methodologies for matching ARGs include energy minimization framework, spectral method, MRF labelling and Bayesian approach.

Energy minimization framework [3] attempts to construct a cost function for matching two ARGs. Minimizing this cost function over all possible vertex matching configurations leads to the distance of two ARGs as the minimum energy. computation is usually approximated by heuristics or random algorithms such as genetic algorithms. Energy minimization framework treats every vertex equally and does not consider the uncertainty of the vertex correspondences between the input graphs, although the attribute extraction process in practice often involves noise contamination. This type of technique also lacks a principled approach to incorporating human knowledge and learning the similarity from training data.

Recently, spectral method has gained attention in the machine learning communities for graph partitioning [9]. Spectral method for ARG matching[8] can be considered a computation scheme for energy minimization formulation. Spectral method is realized by the spectral decomposition of the attribute matrix and constructing the minimum energy solution using eigen vectors. Since researchers have done substantial work on the efficient algorithms for finding eigen vectors, spectral method can achieve efficient computation by taking advantage of those fast algorithms.

Markov Random Field (MRF) labelling [5] is a probabilistic formulation for the ARG matching problem. Essentially, MRF framework converts the energy function used in the energy minimization framework to the Gibbs distribution. Matching problem is thereby reduced to finding the maximum likelihood matching configuration. The similarity measure is finally aggregated from the similarity measure of the individual vertexes by utilizing the labelling results. Since the aggregation process and labelling process performs in separate stages, there is no principled approach to penalize the mismatched vertexes for computing ARG similarity. Moreover, learning potential functions has to be realized by annotating vertex pairs, resulting in expensive human costs.

Bayesian framework proposed by Wilson and Hancock [7] is another probabilistic formulation for the ARG matching problem. Bayesian framework poses the matching problem as calculating and maximizing the Maximum A posteriori Probability (MAP) of the vertex correspondences. Graph similarities is not addressed in the paper, but may be calculated by aggregating the vertex similarities after MAP maximization. The conditional probability involved in MAP calculation is

designed manually without a learning process to adjust the parameters.

There are also other works on topological graph matching without attributes attached to graphs. Topological graph matching is known as graph isomorphism problem if two graphs are of equal sizes and subgraph isomorphism problem if they are not. Subgraph isomorphism problem was proven to be NP-complete, and therefore needs approximate algorithms for efficient computation. One common approach is to reduce the subgraph isomorphism problem to the Maximum Clique problem, which is then solved by heuristic algorithms. Interested readers are referred to [6].

In this report, we establish a novel stochastic framework for matching two ARGs, which defines the ARG similarity as the likelihood or likelihood ratio of the stochastic process that transforms one ARG to the other. The stochastic approach offers a principled way to define the ARG similarity that accounts for both the attribute and topological differences of the two ARGs. More importantly, this framework allows the similarity measure to be learned from training data in both supervised (using vertex pair annotation) and unsupervised manner (using graph pair annotation). The stochastic transformation process is represented as a graphical model on top of the product graph constructed from two input ARGs. Exactly computing the likelihood of the stochastic process is intractable in that we need to sum over all possible vertex correspondence configurations. Yet we take the advantage of the recent development on variational approximation and the Loopy Belief Propagation algorithm to approximate the exact likelihood. The loopy belief propagation is further simplified by taking advantage of the particular topology of the product graph to realize more efficient computation. Unsupervised learning using graph pair annotation is realized by a variational Expectation-Maximization process.

We studied the use of ARG matching technique for part-based visual scene similarity measure. A visual scene is considered as the composition of visual objects or salient parts with specific relations. Such a part-relation model can be represented as an Attributed Relational Graph. Matching two visual scene is thereby reduced to the ARG similarity problem. We apply the part-based similarity model to detect near-Duplicate images in the image database. Promising results are observed by using the proposed ARG matching technique.

2 Stochastic Framework for ARG similarity

Attributed Relational Graph (ARG) is an extension of the ordinary graph by associating discrete or real-valued attributes to its vertexes and edges. The use of attributes allows ARG not only be able to model the topological structure of an entity but also its non-structural properties, which often can be represented as feature vectors. ARG is defined as following

Definition 1 *An attributed relational graph is a triple $G = (V, E, A)$, where V is the vertex set, E is the edge set, and A is the attribute set that contains unary attribute a_i attaching to each node $n_i \in V$, and binary attribute a_{ij} attaching to each edge $e_k = (n_i, n_j) \in E$.*

Figure 1 shows an example of using ARG for part-based modelling in computer vision.

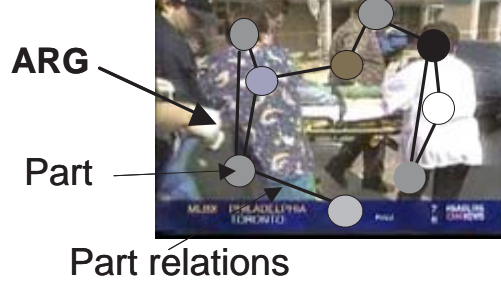


Figure 1: Attributed Relational Graph for Part-based Modelling

In order to match two ARGs, we use a stochastic process to model the transformation from one ARG to the other. The similarity is measured by the likelihood ratio of the stochastic transformation process. We refer such a definition of data similarity as the *transformation principle of similarity*(TPS).

Let H denotes the binary random variable corresponding to two hypotheses relating to the similarity: $H = 1$, the target graph G_t is similar to G_s ; $H = 0$, the two images are not similar. Let $Y^s = \{\{Y_i^s\}; \{Y_{ij}^s\} | i, j \leq N\}$ denotes the unary and binary features of the source graph G_s ; $Y^t = \{\{Y_u^t\}; \{Y_{uv}^t\} | u, v \leq M\}$ the features of the target graph G_s . Then we have the transformation likelihood $p(Y^t | Y^s, H = 1)$, and $p(Y^t | Y^s, H = 0)$ respectively. The similarity between G_s and G_t is then defined as the likelihood ratio

$$S(G^s, G^t) = \frac{p(Y^t | Y^s, H = 1)}{p(Y^t | Y^s, H = 0)} \quad (1)$$

Note that in general $N \neq M$. We decompose the transformation process $p(Y^t | Y^s, H)$ into two steps. The first step copies the vertexes from G_s to G_t and establishes the correspondence between the vertexes of G_s and G_t , referred to as *vertex copy process* (VCP). The second step transforms the attributes of the copied vertexes, referred to as *attribute transformation process*(ATP). This cascade stochastic process is illustrated in Figure 2. The transformation process requires an intermediate variable to specify the correspondences between the vertexes in G_s and G_t . We denote it as X , referred to as *correspondence matrix*, which is a random 0-1 matrix taking value from $\chi = \{0, 1\}^{N \times M}$. $x_{iu} = 1$ means the i th vertex in G_s corresponds to the u th vertex in G_t . This is illustrated in the Figure 3. For the case of one-to-one correspondences of vertexes in G_s and G_t ,

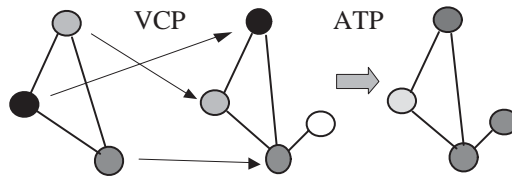


Figure 2: Stochastic Transformation Process for ARG similarity

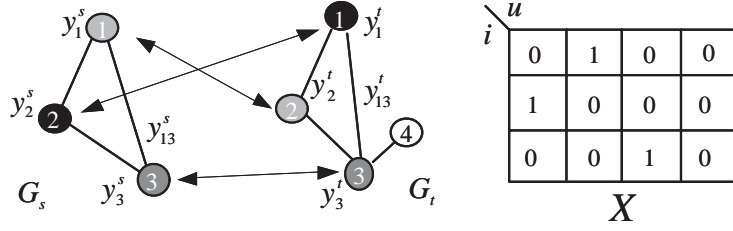


Figure 3: Vertex correspondences of two ARGs and the correspondence matrix

we need to have the following constraints

$$\sum_i x_{iu} \leq 1; \sum_u x_{iu} \leq 1 \quad (2)$$

By introducing X , the transformation process then can be factorized as following:

$$p(Y^t|Y^s, H) = \sum_{X \in \mathcal{X}} p(Y^t|Y^s, X, H)p(X|Y^s, H) \quad (3)$$

$p(X|Y^s, H)$ characterizes the VCP and $p(Y^t|Y^s, X, H)$ represents the ATP. The stochastic transformation process can be represented as a generative graphical model shown in Figure 4. In order to satisfy the constraints in the Eq.(2), we let VCP be represented as a Markov Random Field (MRF) (referred to as prior MRF) with a two-node potential that ensures the one-to-one constraint being satisfied. The MRF model has the following form

$$p(X|Y^s, H = h) = \frac{1}{Z(h)} \prod_{iu, jv} \psi_{iu, jv}(x_{iu}, x_{jv}) \prod_{iu} \phi_h(x_{iu})$$

where $Z(h)$ is the partition function. $h \in \{0, 1\}$ is the hypothesis. Note here we use notations iu and jv as the indices of the elements in the correspondence matrix X and vertexes of the MRF. The MRF model in fact is built upon the product graph of graph G_s and G_t (Figure 5). Each vertex of the product graph is attached with a state variable x_{iu} that signifies one possible matching

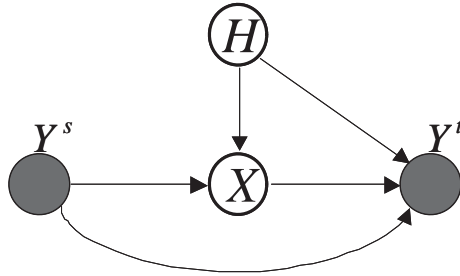


Figure 4: The generative model for ARG matching

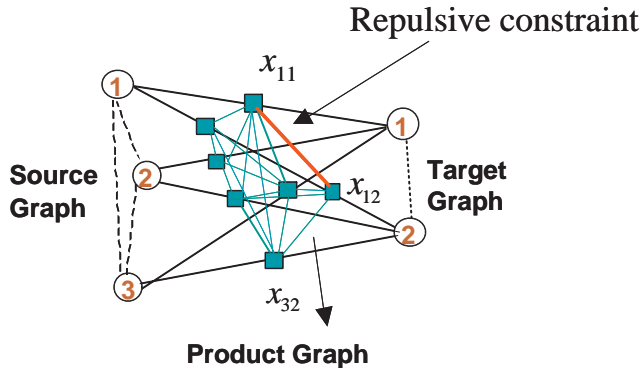


Figure 5: The product graph upon which prior MRF is defined. The repulsive constraint ensures that x_{11} and x_{12} will not equals one simultaneously, i.e. the vertex 1 can only be matched with one vertex in the target graph

between the vertexes in G_s and G_t . For instance, if $x_{iu} = 1$, then the i th vertex in G_s is matched to the u th vertex in G_t . The product graph is a fully connected graph.

For simplifying the model, we assume that X is independent on Y^s given H . The 2-node potential functions are defined as

$$\begin{aligned} \psi_{iu,jv}(0,0) &= \psi_{iu,jv}(0,1) = \psi_{iu,jv}(1,0) = 1, \quad \forall iu,jv \\ \psi_{iu,jv}(1,1) &= 0, \text{ for } i = j \text{ or } u = v; \quad \psi_{iu,jv}(1,1) = 1, \text{ otherwise} \end{aligned}$$

The second line of the above equations is the repulsive constraint to enforce one-to-one correspondence listed in Eq.(2).The 1-node potential function is defined as

$$\phi_0(0) = p_0 \quad \phi_0(1) = q_0; \quad \phi_1(0) = p_1 \quad \phi_1(1) = q_1$$

where p_0, q_0, p_1, q_1 controls the probability that the vertexes in the source graph are copied to the target graph under hypothesis $H = 0, H = 1$. Due to the partition function, any p_h, q_h with identical ratio p_h/q_h would result in the same distribution. Therefore, we can set p_h as 1, and let q_h be learned from training data.

It is not difficult to show that the partition function has the form $Z(h) = \sum_{i=1}^N \binom{N}{i} \binom{M}{i} i! q_h^i$. For efficient learning and likelihood calculation, we use the asymptotic approximation of $Z(h)$ as shown in the following theorem

Theorem 1 *Let $N \leq M$. When $N \rightarrow \infty$, and $M - N < \infty$ The log partition function $\log(Z(h))$ tends to*

$$N[\log(q_h) + c] \tag{4}$$

where c is a constant having the following form

$$c = \lim_{N \rightarrow \infty} \frac{1}{N} \log \left[\sum_{i=1}^N \binom{N}{i} \binom{N}{i} i! \right] \approx 4.1138$$

which is calculated numerically.

Proof of this theorem is realized by representing the partition function as a generalized hypergeometric function ${}_2F_0(\cdot)$, then applying asymptotic analysis on its corresponding Ordinary Differential Equation. The details can be found in the Appendix.

For the *ATP*, we use naïve Bayes assumption to let all unary and binary features independent given Y^s and H . Therefore the *ATP* is fully factorized as

$$p(Y^t|X, Y^s, H) = \prod_{iu, jv} p(y_{uv}^t|x_{iu}, x_{jv}, y_{ij}^s) \prod_{iu} p(y_u^t|x_{iu}, y_i^s)$$

We assume that the Attribute Transformation Process is governed by Gaussian distributions with different parameters. Accordingly, we have conditional density functions for unary attributes

$$\begin{aligned} p(y_u^t|x_{iu} = 1, y_i^s) &= \mathcal{N}(y_i^s, \Sigma_1); \\ p(y_u^t|x_{iu} = 0, y_i^s) &= \mathcal{N}(y_i^s, \Sigma_0) \end{aligned} \quad (5)$$

And conditional density functions for binary attributes

$$\begin{aligned} p(y_{uv}^t|x_{iu} = 1, x_{jv} = 1, y_{ij}^s) &= \mathcal{N}(y_{ij}^s, \Sigma_{11}) \\ p(y_{uv}^t|(x_{iu} \cap x_{jv}) = 0, y_{ij}^s) &= \mathcal{N}(y_{ij}^s, \Sigma_{00}) \end{aligned} \quad (6)$$

The parameters $\Sigma_1, \Sigma_{11}, \Sigma_0, \Sigma_{00}$ are covariance matrices that need to be learned from training data.

3 Induced Markov Random Field

According to these setups, the term inside the equation (3) can be written as

$$\begin{aligned} &p(Y^t|Y^s, X, H = h)p(X|Y^s, H = h) \\ &= \frac{1}{Z(h)} \prod_{ij, uv} \psi'_{iu, jv}(x_{iu}, x_{jv}; y_{ij}^s, y_{uv}^s) \prod_{ij} \phi'_h(x_{iu}, y_i^s, y_u^t) \end{aligned}$$

Where

$$\begin{aligned} \psi'_{iu, jv}(x_{iu}, x_{jv}; y_{ij}^s, y_{uv}^s) &= \psi_{iu, jv}(x_{iu}, x_{jv})p(y_{uv}^t|x_{iu}, x_{jv}, y_{ij}^s) \\ \phi'_h(x_{ij}, y_i^s, y_u^t) &= \phi_h(x_{iu})p(y_u^t|x_{iu}, y_i^s) \end{aligned}$$

Therefore, the transformation likelihood becomes

$$p(Y^t|Y^s, H = h) = Z'(Y^t, Y^s, h)/Z(h) \quad (7)$$

where

$$Z'(Y^t, Y^s, h) = \sum_{X \in \mathcal{X}} \prod_{ij, uv} \psi'_{iu, jv}(x_{iu}, x_{jv}; y_{ij}^s, y_{uv}^s) \prod_{ij} \phi'_h(x_{iu}, y_i^s, y_u^t)$$

which in fact is the partition function of the following induced MRF defined upon the product graph (Figure 5)

$$\begin{aligned} p(X|Y^s, Y^t, H = h) \\ = \frac{1}{Z'(Y^t, Y^s, h)} \prod_{ij, uv} \psi'_{iu, jv}(x_{ij}, x_{uv}; y_{ij}^t, y_{uv}^s) \prod_{ij} \phi'_h(x_{iu}, y_i^s, y_u^t) \end{aligned}$$

The potential function of the induced MRF combines the potential function of the prior MRF and the attribute density functions. From these derivation, we therefore reached the following important conclusion

Proposition 1 *The ARG transformation likelihood is the ratio of the partition function of the induced MRF and the partition function of the prior MRF, i.e.*

$$p(Y^t|Y^s, H = h) = \frac{Z'(Y^t, Y^s, h)}{Z(h)} \quad (8)$$

The partition function $Z(h)$ can be approximately computed from Eq.(4). The following section provides the approximate computation scheme for $Z'(Y^t, Y^s, h)$.

4 Computing the Transformation Likelihood

The computation of the exact induced partition function $Z'(Y^t, Y^s, h)$ is intractable, since we have to sum over all possible X in the set χ , whose cardinality grows exponentially with respect to NM . Therefore we have to compute the likelihood using certain approximation.

4.1 Approximation of the Exact Likelihood

By applying the Jensen's inequality on the log partition function $\log Z'(Y^t, Y^s, h)$, we can find its variational lower bound as following. The lower bound can be used for approximating the partition function of the induced MRF

Using short-hand

$$l(X; Y^s, Y^t, h) = \prod_{ij, uv} \psi'_{iu, jv}(x_{iu}, x_{jv}; y_{ij}^t, y_{uv}^s) \prod_{ij} \phi'_h(x_{iu}, y_i^s, y_u^t)$$

and introducing an approximate probability distribution function $q(X)$ over χ , we have

$$\begin{aligned}
\log Z' &= \log \sum_X l(X; Y^s, Y^t, h) \\
&= \log \sum_X q(X) \frac{l(X; Y^s, Y^t, h)}{q(X)} \\
&\geq \max_{q(X)} \sum_X q(X) \log \frac{l(X; Y^s, Y^t, h)}{q(X)} \\
&= \max_{q(X)} \sum_X q(X) \log l(X; Y^s, Y^t, h) + \mathcal{H}(q(X)) \\
&= \sum_X \hat{q}(X) \log l(X; Y^s, Y^t, h) + \mathcal{H}(\hat{q}(X)) \\
&= \sum_{iu, jv} \hat{q}(x_{iu}, x_{jv}) \log \psi'_{iu, jv}(x_{iu}, x_{jv}; y_{ij}^s, y_{uv}^t) \\
&\quad + \sum_{iu} \hat{q}(x_{iu}) \log \phi'_h(x_{iu}, y_i^s, y_u^t) + \mathcal{H}(\hat{q}(X)) \tag{9}
\end{aligned}$$

where

$$\hat{q}(X) = \operatorname{argmax}_{q(X)} \sum_X q(X) \log l(X; Y^s, Y^t, h) + \mathcal{H}(q(X)) = \operatorname{argmin}_{q(X)} KL(q(X) || l(X))$$

is the best approximated probability distribution function in the sense of KL divergence $KL(\hat{q}(X) || l(X))$. $\hat{q}(x_{iu})$ and $\hat{q}(x_{iu}, x_{jv})$ are the one-node and two-node marginals of $\hat{q}(X)$, also known as beliefs. $\mathcal{H}(\hat{q}(X))$ is the entropy of $\hat{q}(X)$, which does not have tractable decomposition for loopy graphical model (here our MRF is fully connected graph). Yet the entropy $\mathcal{H}(\hat{q}(X))$ can be approximated using Bethe/Kikuchi approximation [10], by which [10] has shown that the $\hat{q}(x_{iu})$ and $\hat{q}(x_{iu}, x_{jv})$ can be obtained by a set of fixed point equations, known as Loopy Belief Propagation (LBP). The entropy for the Bethe approximation in our case is

$$\mathcal{H}(\hat{q}(X)) = - \sum_{(iu, jv)} \sum_{(x_{iu}, x_{jv})} \hat{q}(x_{iu}, x_{jv}) \log \hat{q}(x_{iu}, x_{jv}) + \sum_{iu} (MN - 2) \sum_{x_{iu}} \hat{q}(x_{iu}) \log(\hat{q}(x_{iu}))$$

4.2 Modified Loopy Belief Propagation

The speed bottleneck of this algorithm is the Loopy Belief Propagation. Without any speedup, the complexity of BP is $O((N \times M)^3)$, which is formidable for computation. However, for fully connected graph, the complexity of BP can be reduced by introducing an auxiliary variable. The BP message update rule is modified as follows

$$\begin{aligned}
m_{iu, jv}^{(t+1)}(x_{jv}) &= k \sum_{x_{iu}} \psi_{iu, jv}(x_{iu}, x_{jv}) \phi_h(x_{iu}) M_{iu}^{(t)}(x_{iu}) / m_{jv, iu}^{(t)}(x_{iu}) \\
M_{iu}^{(t+1)}(x_{iu}) &= \exp \left(\sum_{kw \neq iu} \log(m_{kw, iu}^{(t+1)}(x_{iu})) \right)
\end{aligned}$$

Where M_{iu} is the introduced auxiliary variable, t is the iteration index. After the message update converges, the one node and two node beliefs $\hat{q}(x_i)$ and $\hat{q}(x_{iu}, x_{jv})$ can be computed as following

$$\begin{aligned}\hat{q}_{iu}(x_{iu}) &= k\phi_h(x_{iu})M_{iu}(x_{iu}) \\ \hat{q}(x_{iu}, x_{jv}) &= k\psi_{iu,jv}(x_{iu}, x_{jv})\phi_h(x_{iu})\phi_h(x_{jv})M_{iu}(x_{iu})M_{jv}(x_{jv})/(m_{jv,iu}(x_{iu})(m_{iu,jv}(x_{jv})))\end{aligned}$$

Where k is the normalization constant. This modification results in $O((N \times M)^2)$ computation.

To further speed up the likelihood computation, early determination of the node correspondences (i.e. x_{iu}) before LBP can be conducted by removing the vertexes whose potential function is above certain threshold. This operation reduces the vertex number of the induced MRF model, resulting in less computation cost. Another speed-up scheme may be to use naive mean field approximation [4] instead of Bethe approximation. However, its accuracy may be worse than that of Bethe approximation.

5 Learning the Parameters

Learning ARG matching can be performed at two levels: vertex-level and graph-level. For the vertex level, the annotators annotate the correspondence of every vertex pair. This process is very expensive, since in computer vision application, an image typically has 50-200 vertexes. In order to reduce human supervision, graph-level learning can be used, where annotators only indicate whether two ARGs are similar or not without identifying specific corresponding vertexes .

For learning at the vertex-level, maximum likelihood estimation results in the direct calculation of the mean and variance of Gaussian functions in the Eq.(5)(6) from the training data. The resulting estimates of the parameters are thereby used as the initial parameters for the graph-level learning.

For learning at the graph-level, we can use the following variational Expectation-Maximization (E-M) scheme:

E Step: Compute $\hat{q}(x_{iu})$ and $\hat{q}(x_{iu}, x_{jv})$ using Loopy Belief Propagation.

M Step: Maximize the lower bound in Eq.(9) by varying parameters. This can be realized by differentiating the lower bound with respect to the parameters, resulting in the following update

equations

$$\begin{aligned}
\xi_{iu} &= \hat{q}(x_{iu} = 1); & \xi_{iu,jv} &= \hat{q}(x_{iu} = 1, x_{jv} = 1) \\
\Sigma_1 &= \frac{\sum_k \sum_{iu} (y_u^t - y_i^s)(y_u^t - y_i^s)^T \xi_{iu}}{\sum_k \sum_{iu} \xi_{iu}} \\
\Sigma_0 &= \frac{\sum_k \sum_{iu} (y_u^t - y_i^s)(y_u^t - y_i^s)^T (1 - \xi_{iu})}{\sum_k \sum_{iu} (1 - \xi_{iu})} \\
\Sigma_{11} &= \frac{\sum_k \sum_{iu,jv} (y_{uv}^t - y_{ij}^s)(y_{uv}^t - y_{ij}^s)^T \xi_{iu,jv}}{\sum_k \sum_{iu,jv} \xi_{iu,jv}} \\
\Sigma_{00} &= \frac{\sum_k \sum_{iu,jv} (y_{uv}^t - y_{ij}^s)(y_{uv}^t - y_{ij}^s)^T (1 - \xi_{iu,jv})}{\sum_k \sum_{iu,jv} (1 - \xi_{iu,jv})}
\end{aligned}$$

Where k is the index for the instances of the training graph pairs.

For the prior parameter q_h , the larger ratio q_1/q_0 would result in the more contribution of the prior model to the overall transformation likelihood ratio. Currently, we use a gradient descent scheme to gradually increase the value of q_1 (start from 1) until we achieve the optimal discrimination of IND classification in the training data. The discriminative criterion function is defined in the following equation

$$D(q_1) = \left[\sum_{i \in T^+} S(G_i^s, G_i^t) - \sum_{j \in T^-} S(G_j^s, G_j^t) \right] / \sum_{i \in T^+} S(G_i^s, G_i^t)$$

Where T^+ is the set consisting of positive training pairs, and T^- is the set consisting of negative training pairs.

6 Application to Image Near-Duplicate Detection

In order to test the algorithm, we apply the ARG matching technique to find Image Near-Duplicate (IND) in video database. Image Near-Duplicate refers to two images that are close to exact duplicate but with variations due to content changes, camera parameter changes, and digitization conditions. Figure 6 shows several examples of IND.

The transformation between one image to the other in IND usually cannot be accommodated by linear transform such as affine transformation. We represent the image scene using ARG based modelling, where each part of the visual scene corresponds to one vertex in ARG. The parts are extracted by using interest point detector (SUSAN corner detector), afterwards local descriptor around the interest points are calculated to represent the appearance of the part. We use an eleven feature vector consisting of RGB colors, spatial locations and Gabor filter coefficients. There are also relational features being defined at the edges of the ARGs. We use the spatial coordinate differences to form two dimensional feature vectors for relational features defined at edges. Currently, all ARGs are fully connected graphs.



Figure 6: Examples of Image Near-Duplicate

6.1 Implementation Issues of ARG matching

From the approximate likelihood equations, we observe that the likelihood ratio is not invariant to the sizes of the input graphs. To reduce the sensitivity of the size variation. We use the averaged likelihood ratio $S(G^s, G^t)/(MN)$ instead of $S(G^s, G^t)$. Furthermore, we assume that under negative hypothesis $H = 0$, there is no vertex correspondence between G_s and G_t . Therefore, all x_{iu} and $x_{iu,jv}$ are set to 0 with probability 1. And the Gaussian parameters for $H = 0$ are the same as those for $H = 1$. To reduce the computational cost, we use the early determination scheme by a thresholding process. As a result, the maximum size of the induced MRF is 150 vertexes. The average computation time of matching two images is about 0.4 second. However, if two images are Near-Duplicate, the computation time could be as high as 60 seconds. The speed may be further boosted using more sophisticated fast Belief Propagation schemes.

6.2 Experiments of Image Near-Duplicate Detection

The benchmark data are collected from the TREC-VID 2003 corpus. TREC-VID is an open benchmark for evaluating visual concept (objects, scenes and events) detection and semantic video retrieval. TREC-VID 2003 consisting of news videos broadcasted from January 1998 to June 1998. The news videos are from two major US broadcast channels: CNN and ABC.

The IND detection data set consists of 150 IND pairs (300 images) and 300 non-duplicate images extracted from TREC-VID 2003 video frames. The entire data set is partitioned into training set and testing set. The training set consists of 30 IND pairs and 60 non-duplicate images.

Learning process consists of two phases. In the first phase, we apply the supervised vertex-level learning, where the correspondences of interest points are marked by the annotator. The number of interest points varies from 50 to 200 per image. Only 5 near-duplicate pairs and 5 non-duplicate pairs are used in vertex-level learning. In the second phase, we conduct the graph-level learning. E-M scheme is carried out using 25 IND pairs and 25 Non-IND pairs.

The performance of the developed method (GRAPH) is compared with color histogram (CH), local edge descriptor (LED), averaged features distance of interest points (AFDIP) and graph matching with manual parameter setting (GRAPH-M). Local edge descriptor has been demon-

strated as the best feature for image copy detection in the previous work [2]. For color histogram, we use HSV color space with 64 bins of H, 8 bins of S, and 4 bins of V. AFDIP is the summation of all possible cosine distances between the unary feature vectors of the interest points in G_s and G_t divided by NM . GRAPH-M is the graph matching likelihood ratio under the manually selected Gaussian parameters. The parameters are obtained by manually adjusting the covariance matrices until we observe the best interest point matching results (by binarizing the belief $\hat{q}(x_{iu})$). The parameters are initialized using vertex-level learning with two IND pairs. *recall* is defined as the number of the correctly detected IND pairs divided by the number of all ground truth IND pairs. *Precision* is defined as the number of the correctly detected IND pairs divided by the number of all detected IND pairs. The results are shown in figure 7.

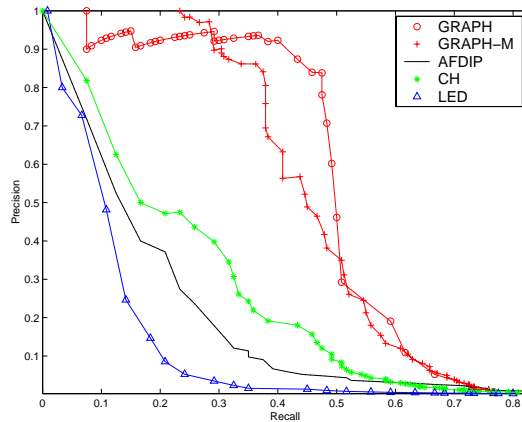


Figure 7: The performance of IND detection using ARG matching

7 Conclusion

We presented a stochastic framework for calculating the similarity of two Attributed Relational Graphs. Such a stochastic framework offers a principled definition of ARG similarity and allows the similarity to be learned from training data in supervised or unsupervised manner. Future work is to further reduce the computational complexity of the matching process using different variational approximation schemes.

8 Appendix

We provide the proof of theorem 1 as following.

We note that the partition function $Z(h)$ can be represented as a generalized hypergeometric function

$$Z(h) = \sum_{i=0}^N \binom{N}{i} \binom{M}{i} i! z^i = {}_2F_0(-N; -M; ; z) \quad (10)$$

Here we use z to substitute q_h . ${}_2F_0(a; b; ; z)$ is a generalized hypergeometric function [1]:

$${}_2F_0(a; b; ; z) = \sum_{n=0}^{\infty} (a)_n (b)_n \frac{z^n}{n!} \quad (11)$$

where $(a)_p, (b)_p$ are rising factorials, or Pochhammer symbols, with $(a)_p = a(a+1)\cdots(a+p)$. ${}_2F_0(a; b; ; z)$ is the solution of the following Ordinary Differential Equation (ODE):

$$z^2 y'' + [(a+b+1)z-1]y' + aby = 0 \quad (12)$$

Therefore the partition function is the solution of the following ODE:

$$z^2 y'' + [(1-N-M)z-1]y' + MNy = 0 \quad (13)$$

To solve it, we make a change of variable to let $y = e^{Nw}$, then we have

$$\begin{aligned} y' &= e^{Nw} Nw' \\ y'' &= Nw'' e^{Nw} + N^2 (w')^2 e^{Nw} \end{aligned}$$

plug into Eq. (13), we get

$$z^2 Nw'' + z^2 N^2 (w')^2 + (1-N-M)zNw' + MN = 0$$

Devide the above equation by N^2 , then when $N \rightarrow \infty$ and $M \rightarrow \infty$, the 2nd-order ODE degenerates to the following 1st-order ODE

$$z^2 (w')^2 - 2zw' + 1 = 0$$

which gives $w' = \pm \frac{1}{z}$. But since y has to be monotonically increasing, w' has to be positive. Therefore we have solution $w = \ln(z) + \lambda$, where λ is a constant. Accordingly, we have

$$y = e^{N\lambda} z^N$$

To obtain the constant λ , we let $z = 1$, then

$$\lambda = \lim_{N \rightarrow \infty} \frac{1}{N} \log \left[\sum_{i=1}^N \binom{N}{i} \binom{N}{i} i! \right]$$

Therefore, when $N \rightarrow \infty$, the log partition function tends to

$$N[\ln(z) + \lambda]$$

□

References

- [1] H. Exton. Handbook of hypergeometric integrals: Theory, applications, tables, computer programs. *Chichester, England: Ellis Horwood*, 1978.
- [2] A. Hampapur. Comparison of distance measures for video copy detection. In *Proceedings of ICME 2001*, pages 188–192. IEEE, August 2001.
- [3] H.G.Barrow and R.J. Popplestone. Relational descriptions in picture processing. *Machine Intelligence*, 6:377–396, 1971.
- [4] T. Jaakkola. Tutorial on variational approximation methods. In *Advanced mean field methods: theory and practice. MIT Press, 2000.*, 2000.
- [5] Stan Z. Li. Mrf modeling in computer vision. *Springer-Verlag*, 1995.
- [6] M. Pelillo. Replicator equations, maximal cliques, and graph isomorphism. *Neural Computation*, pages 1933 – 1955.
- [7] R.C.Wilson and E.R.Hancock. Structural matching by discrete relaxation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 19:1–2, 1997.
- [8] J. Shi and J. Malik. Self inducing relational distance and its application to image segmentation. *Lecture Notes in Computer Science*, 1406:528, 1998.
- [9] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.
- [10] J S Yedidia and W T Freeman. Understanding belief propagation and its generalizations. In *Exploring Artificial Intelligence in the New Millennium*, pages Chap. 8, pp. 239–236, January 2003.