Clustering Graphs by Weighted Substructure Mining

Koji Tsuda KOJI.TSUDA@TUEBINGEN.MPG.DE
Max Planck Institute for Biological Cybernetics, Spemannstr. 38, 72076 Tübingen, Germany

Taku Kudo TAKU@GOOGLE.COM
Google Japan Inc., Cerulean Tower 6F, 26-1 Sakuragaoka-cho, Shibuya-ku, Tokyo, 150-8512, Japan

Abstract

Graph data is getting increasingly popular in, e.g., bioinformatics and text processing. A main difficulty of graph data processing lies in the intrinsic high dimensionality of graphs, namely, when a graph is represented as a binary feature vector of indicators of all possible subgraphs, the dimensionality gets too large for usual statistical methods. We propose an efficient method for learning a binomial mixture model in this feature space. Combining the ℓ_1 regularizer and the data structure called DFS code tree, the MAP estimate of non-zero parameters are computed efficiently by means of the EM algorithm. Our method is applied to the clustering of RNA graphs, and is compared favorably with graph kernels and the spectral graph distance.

1. Introduction

Graphs are general and powerful data structures that can be used to represent diverse kinds of objects. Much of the real world data is represented not as vectors, but as graphs including sequences and trees, for example, biological sequences, semi-structured texts such as HTML and XML, chemical compounds, RNA secondary structures, and so forth.

To derive useful knowledge from the graph database, a possible first step is to partition the objects into subgroups using *clustering* techniques (McLachlan & Basford, 1998). So far, a number of graph classification methods have been proposed. See (Wilson et al., 2005) for an extensive review of this subject. They are based either on the alignment of graphs (Sanfeliu & Fu, 1983) or comparison of features derived from graphs (Kashima et al., 2003). Since most alignment-based techniques are only applicable to small datasets, the feature-based techniques are more promising for practical applications. There are many possible ways to derive features from a graph, but one of the most natural way would be to represent a graph by a set of its *subgraphs*. In fact, in classifying chemical compounds, the set of small graphs (i.e., *patterns*) are determined a priori, and a graph corresponding to a molecule is represented by a feature vector of binary indicators of patterns (Gasteiger & Engel, 2003). Alternatively, one can use simpler but less informative features such as label paths (Kashima et al., 2003) or the spectrum of Laplacian matrices (Wilson et al., 2005) for fast computation.

For general applications with insufficient domain knowledge, it is difficult to select the patterns manually in advance. To select them automatically, one possible approach is to use *frequent substructure mining methods* (Inokuchi et al., 2000; Yan & Han, 2002) to find the set of patterns that appear frequently in the database. However, frequently appearing patterns are not necessarily useful for clustering.

In this paper, we propose a graph clustering method that selects informative patterns at the same time. In principle, our task is analogous to feature selection for vectors (Roth & Lange, 2004), however, the difference is that the features (i.e., patterns) are not explicitly listed. For vectors, it is often practised that a certain score function is evaluated on each feature and those with high scores are chosen. However, since the number of possible patterns grows exponentially to graph size, it is computationally prohibitive to evaluate the scores for all possible patterns. To realize the fast search of high scoring patterns, a tree-shaped search space is constructed (Figure 1). Each node of the tree has a pattern and the tree is organized such that child

Appearing in *Proceedings of the 23rd International Conference on Machine Learning*, Pittsburgh, PA, 2006.

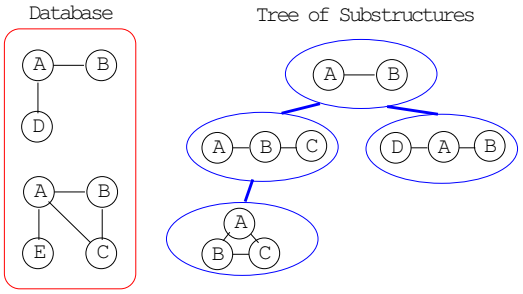

Figure 1. Schematic figure of the tree-shaped search space of patterns (i.e., substructures)

nodes have supergraphs of the parent node. Starting from a set of small patterns, the tree is expanded by generating a child node and adding it to an existing node. Due to the bound of the score function, we do not need to generate the whole tree and the pattern selection is done in a short time.

Our method is fully probabilistic, adopting a binomial mixture model defined on a very high dimensional vector indicating the presence or absence of all possible patterns. A (local) MAP estimate is derived by the EM algorithm. It will be shown how a graph mining algorithm is embedded in the framework of probabilistic inference. In experiments, our method is favorably compared with the marginalized graph kernel by Kashima et al. (2003) and the spectral method by Wilson et al. (2005).

The rest of the paper is organized as follows. In Section 2, we review the graph mining technique used in our clustering algorithm. In Section 3, our graph clustering algorithm is presented, where the graph mining algorithm is combined with the EM algorithm. In Section 4, the experimental results about clustering RNA graphs (Gan et al., 2003) are shown. We conclude this paper in Section 5.

2. Weighted Substructure Mining

Given a database of graphs, the frequent substructure mining algorithms such as AGM (Inokuchi et al., 2000) and gspan (Yan & Han, 2002) enumerate all patterns included in more than m graphs. The threshold m is called *minimum support*. For extracting patterns for clustering, it is needed to extend this framework by introducing weights to characterize the difference among graph clusters (i.e., *weighted substructure mining*). Due to space restriction, we will explain basics and leave details to (Kudo et al., 2005) and references therein.

In this paper, we deal with undirected, labeled and connected graphs. To be more precise, we define the graph and its subgraph as follows:

Definition 1 (Labeled connected graph). *A labeled graph is represented in a 4-tuple $G = (V, E, \mathcal{L}, l)$, where V is a set of vertices, $E \subseteq V \times V$ is a set of edges, $\mathcal{L}$ is a set of labels, and $l : V \cup E \to \mathcal{L}$ is a mapping that assigns labels to the vertices and edges. A labeled connected graph is a labeled graph such that there is a path between any pair of vertices.*

Definition 2 (Subgraph). *Let $G' = (V', E', \mathcal{L}', l')$ and $G = (V, E, \mathcal{L}, l)$ be labeled connected graphs. G' is a subgraph of G ($G' \subseteq G$) if the following conditions are satisfied: (1) $V' \subseteq V$, (2) $E' \subseteq E$, (3) $\mathcal{L}' \subseteq \mathcal{L}$, and (4) $l = l'$. If G' is a subgraph of G, then G is a supergraph of G'.*

Given a graph database $\mathcal{G} = \{G_i\}_{i=1}^n$, let $\mathcal{T} = \{T_k\}_{k=1}^d$ be the set of all patterns, i.e., the set of all subgraphs included in at least one graph in $\mathcal{G}$. Then, each graph G_i is encoded as a d-dimensional vector $\boldsymbol{x}_i$,

$$x_{ik} = I(T_k \subseteq G_i),$$

where $I(\cdot)$ is 1 if the condition inside is true and 0 otherwise. In frequent substructure mining, the task is to enumerate all patterns whose support is more than m,

$$S_{freq} = \{k \mid \sum_{i=1}^{n} x_{ik} \geq m\}. \tag{1}$$

However, in clustering, we do not need frequent subgraphs which appear in every cluster. In (Kudo et al., 2005), the weight vector $\boldsymbol{w} \in \Re^n$ are introduced and the following set of subgraphs are derived,

$$S_{\boldsymbol{w}} = \{k \mid \left|\sum_{i=1}^{n} w_i(2x_{ik} - 1)\right| \geq \tau\}, \tag{2}$$

where τ is a predetermined constant. By setting the weights positive in one cluster and negative in the other cluster, one can extract discriminative patterns whose frequencies are significantly different in the two clusters.

Since our method deals with more than two clusters, multiple weight vectors $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_c$ and thresholds $\tau_1, \ldots, \tau_c$ are introduced, and the set of discriminative patterns is obtained as $S_W = S_{\boldsymbol{w}_1} \cup \cdots \cup S_{\boldsymbol{w}_c}$. Equivalently, S_W is rewritten as

$$S_W = \{k \mid \max_{\ell=1,\ldots,c} \left|\sum_{i=1}^{n} w_{\ell i}(2x_{ik} - 1)\right| - \tau_\ell \geq 0\}. \tag{3}$$

Efficient enumeration of S_W is done using the DFS code tree as explained in the next subsection.

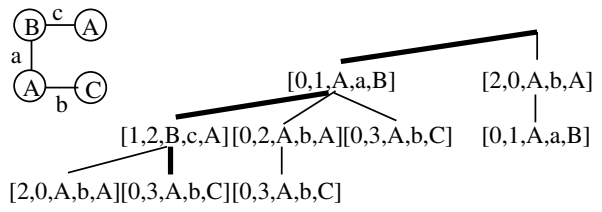

Figure 2. Example of the DFS code tree. A pattern is represented as a path from the root to a node. For example, the highlighted path corresponds to the graph shown above.

2.1. DFS Code Tree

The key idea of efficient graph mining is to exploit the *anti-monotonicity*, namely the frequency of a pattern is always smaller than or equal to that of its subgraph,

$$T_j \subseteq T_k \;\Rightarrow\; \sum_{i=1}^{n} x_{ij} \geq \sum_{i=1}^{n} x_{ik}.$$

In frequent substructure mining (1), one constructs a tree-shaped search space where each node corresponds to a pattern (Figure 1). The tree is generated from the root with an empty graph, and the pattern of a child node is made by adding one edge. As the pattern gets larger, the frequency decreases monotonically. If the frequency of the generated pattern T_k is less than m, it is guaranteed that the frequency of any supergraph of T_k is also less than m. Therefore, the exploration is stopped there (i.e., *tree pruning*). By repeating node generation until all possibilities are checked, all frequent subgraphs are enumerated.

Actual implementation of the search tree is different among mining methods, but we adopted the DFS (depth first search) code tree used in the gspan algorithm (Figure 2). Each node corresponds to an edge represented as a 5-tuple (i, j, v_i, e_{ij}, v_j), where i, j are the node indices and e_{ij}, v_i, v_j are the labels of $i-j$ edge, i-th vertex and j-th vertex, respectively. Also, each node maintains a set of pointers to the corresponding edges of graphs in database. A pattern is represented as a path from the root to a node, avoiding the redundancy of storing similar patterns.

In the tree expansion process, it often happens that the generated pattern is isomorphic to one of the patterns that have already been generated. It leads to significant loss of efficiency because the same pattern is checked multiple times. The gspan algorithm solves this problem by the minimum DFS code approach, and we also adopted it for pruning isomorphic patterns.

In weighted substructure mining (3), the search tree is pruned by a different condition. Let us rewrite the weight as $w_{\ell i} = y_{\ell i} d_{\ell i}$ where $d_{\ell i} = |w_{\ell i}|$ and $y_{\ell i} = \mathrm{sign}(w_{\ell i})$. Then, the following bound is obtained: For any $T_j \subseteq T_k$,

$$\left| \sum_{i=1}^{n} w_{\ell i} (2x_{ik} - 1) \right| \leq \gamma_\ell,$$

where $\gamma_\ell = \max(\gamma_\ell^+, \gamma_\ell^-)$ and

$$\gamma_\ell^+ = 2 \sum_{\{i | y_{\ell i} = +1, T_j \subseteq G_i\}} d_{\ell i} - \sum_{i=1}^{n} d_{\ell i} y_{\ell i},$$

$$\gamma_\ell^- = 2 \sum_{\{i | y_{\ell i} = -1, T_j \subseteq G_i\}} d_{\ell i} + \sum_{i=1}^{n} d_{\ell i} y_{\ell i}.$$

See (Kudo et al., 2005) for the proof. Summarizing the bounds, we get the following bound for weighted substructure mining (3),

$$\max_{\ell=1,\cdots,c} |\sum_{i=1}^{n} w_{\ell i}(2x_{ik} - 1)| - \tau_\ell \leq \max_{\ell=1,\cdots,c} (\gamma_\ell - \tau_\ell). \quad (4)$$

When a pattern T_j is generated, the scores of its supergraphs T_k are upperbounded as above. Thus, if the upperbound is negative, we can safely quit further exploration.

3. Binomial Mixture Model for Graphs

In this section, we will present an efficient method for learning a binomial mixture model on the very high dimensional binary vector $\boldsymbol{x}$. The weight substructure mining (3) will be used in the EM algorithm to avoid the computational difficulty.

To begin with, let us review a typical clustering method using a binomial mixture model,

$$p(\boldsymbol{x}|\Theta) = \sum_{\ell=1}^{c} \alpha_\ell p_\ell(\boldsymbol{x}|\boldsymbol{\theta}_\ell),$$

where $p_\ell(\boldsymbol{x}|\boldsymbol{\theta}_\ell) = \prod_{k=1}^{d} p_{\ell k}(x_{\ell k}|\theta_{\ell k})$ and each binomial distribution is parametrized as

$$p_{\ell k}(x_k|\theta_{\ell k}) = \frac{\exp(\theta_{\ell k} x_k)}{1 + \exp(\theta_{\ell k})}. \quad (5)$$

When n graphs in the database are encoded as $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$, the log likelihood is described as

$$\operatorname*{argmax}_{\Theta} \sum_{i=1}^{n} \log \sum_{\ell=1}^{c} \alpha_\ell p_\ell(\boldsymbol{x}_i|\boldsymbol{\theta}_\ell). \quad (6)$$

We assume the mixture ratio α_ℓ is known for simplicity. The parameters are optimized by the EM algorithm:

- (E-step) Given $\theta_{\ell k}$, the posterior probability $r_{\ell i} = p(y = \ell | \boldsymbol{x}_i, \Theta)$ is computed as

$$r_{\ell i} = p(y = \ell | \boldsymbol{x}_i, \Theta) = \frac{\alpha_\ell p_\ell(\boldsymbol{x}_i | \boldsymbol{\theta}_\ell)}{\sum_\ell \alpha_\ell p_\ell(\boldsymbol{x}_i | \boldsymbol{\theta}_\ell)}. \quad (7)$$

- (M-step) Given $r_{\ell i}$, the complete data likelihood

$$\frac{1}{n} \sum_{i=1}^{n} \sum_{\ell=1}^{c} r_{\ell i} \sum_{k=1}^{d} \log p_{\ell k}(x_{ik} | \theta_{\ell k}) \quad (8)$$

is maximized with respect to $\theta_{\ell k}$.

After the convergence of the EM algorithm, clustering is done by classifying each vector according to its posterior probabilities.

3.1. Regularization

Since the dimensionality d is extremely large, it is computationally prohibitive to carry out the sum over all features in both E and M steps. To cope with this problem, let us introduce a set of *baseline constants* θ_{0k}, and regularize the likelihood as

$$\frac{1}{n} \sum_{i=1}^{n} \log \sum_{\ell=1}^{c} \alpha_\ell p_\ell(\boldsymbol{x}_i | \boldsymbol{\theta}_\ell) - \lambda \sum_{\ell=1}^{c} \sum_{k=1}^{d} |\theta_{\ell k} - \theta_{0k}|, \quad (9)$$

where the parameters $\theta_{\ell k}$ are attracted to the constant θ_{0k}. The M-step is modified as

$$\frac{1}{n} \sum_{i=1}^{n} \sum_{\ell=1}^{c} r_{\ell i} \sum_{k=1}^{d} \log p_{\ell k}(x_{ik} | \theta_{\ell k}) - \lambda |\theta_{\ell k} - \theta_{0k}|, \quad (10)$$

while the E-step stays the same. Due to sparsity induced by the regularizer, most $\theta_{\ell k}$ will be exactly equal to θ_{0k}. Let F be the set of *active patterns*, namely

$$F = \{k \mid \text{there exists } \ell \text{ such that } \theta_{\ell k} \neq \theta_{0k}\}.$$

Then, the E-step can be computed only with F,

$$p(y = \ell | \boldsymbol{x}) = \frac{\alpha_\ell \prod_{k \in F} p_{\ell k}(x_k | \theta_{\ell k})}{\sum_\ell \alpha_\ell \prod_{k \in F} p_{\ell k}(x_k | \theta_{\ell k})}. \quad (11)$$

Furthermore, the computation of the M-step is made possible, when the baseline constant $\boldsymbol{\theta}_0$ is set to the maximum likelihood estimate from all the graphs, i.e.,

$$\theta_{0k} = \log \eta_{0k} - \log(1 - \eta_{0k}), \quad (12)$$

where η_{0k} is the occurence probability of pattern k,

$$\eta_{0k} = \frac{1}{n} \sum_{i=1}^{n} x_{ik}. \quad (13)$$

Algorithm 1 Pseudo-code of the EM algorithm for clustering graphs.

Set initial partition $r_{\ell i}$ randomly.
repeat
  Set the weight parameter $w_{\ell i}$
  Call the mining algorithm to obtain F
  Estimate $\theta_{\ell k}$ and θ_{0k} only for $k \in F$ (M-step)
  Update the posterior $r_{\ell i}$ (E-step)
until the convergence of (21)

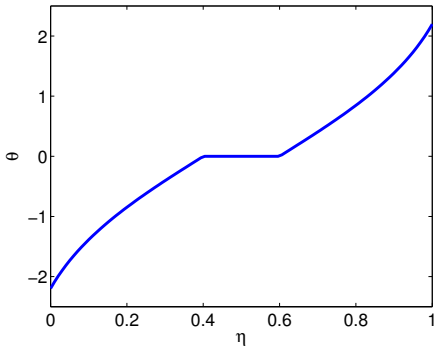

Figure 3. Example of the solution (17), where $\eta_{0k} = 0.5$ and $\lambda_\ell = 0.1$.

3.2. M-step and Substructure Mining

In the M-step (10), each parameter can be solved separately,

$$\min_{\theta_{\ell k}} -\frac{1}{n} \sum_i r_{\ell i} \log p(x_{ik} | \theta_{\ell k}) + \lambda |\theta_{\ell k} - \theta_{0k}|. \quad (14)$$

Furthermore, the solution of this one-dimensional problem can be obtained in a closed form. Let $\eta_{\ell k}$ denote the occurence probability of pattern k within cluster ℓ,

$$\eta_{\ell k} = \sum_i r_{\ell i} x_{ik} / \sum_j r_{\ell j}. \quad (15)$$

Setting $\lambda_\ell = \frac{\lambda n}{\sum_j r_{\ell j}}$, (14) is written as

$$\min_{\theta_{\ell k}} -\eta_{\ell k} \theta_{\ell k} + \log(1 + \exp(\theta_{\ell k})) + \lambda_\ell |\theta_{\ell k} - \theta_{0k}|. \quad (16)$$

Theorem 1. *The solution of (16) is*

$$\theta_{\ell k} = \begin{cases} \log \frac{\eta_{\ell k} - \lambda_\ell}{1 - (\eta_{\ell k} - \lambda_\ell)} & (\eta_{\ell k} \geq \eta_{0k} + \lambda_\ell). \\ \theta_{0k} & (\eta_{0k} - \lambda_\ell \leq \eta_{\ell k} \leq \eta_{0k} + \lambda_\ell) \\ \log \frac{\eta_{\ell k} + \lambda_\ell}{1 - (\eta_{\ell k} + \lambda_\ell)} & (\eta_{\ell k} \leq \eta_{0k} - \lambda_\ell). \end{cases} \quad (17)$$

See Figure 3 for an example of the solution. The proof is deferred to Appendix.

Let F_ℓ be the index set of parameters that are not identical with θ_{0k},

$$F_\ell = \{k \mid \theta_{\ell k} \neq \theta_{0k}\}.$$

Then, $F = F_1 \cup \cdots \cup F_c$. Due to the above theorem, we can identify F_ℓ from the occurence probabilities η_{0k} and $\eta_{\ell k}$. Namely, F_ℓ is equivalently written as

$$F_\ell = \{k \mid |\eta_{\ell k} - \eta_{0k}| \geq \lambda_\ell \}. \tag{18}$$

Substituting (13) and (15) into (18), F_ℓ is further rewritten as

$$F_\ell = \{k \mid \left| \sum_{i=1}^{n} w_{\ell i}(2x_{ik} - 1) \right| \geq 2\lambda_\ell\}$$

where

$$w_{\ell i} = \frac{r_{\ell i}}{\sum_j r_{\ell j}} - \frac{1}{n}.$$

Thus the combined set F is described as

$$F = \{k \mid \max_{\ell=1,\ldots,c} \left| \sum_{i=1}^{n} w_{\ell i}(2x_{ik} - 1) \right| - 2\lambda_\ell \geq 0\}. \tag{19}$$

Here, it turns out that the pattern set S_W of weighted substructure mining (3) coincides with F if $\tau_\ell = 2\lambda_\ell$. Therefore, the set of active patterns F can be enumerated by employing the mining algorithm. In the M-step, we need to calculate the parameters $\theta_{\ell k}$ for $k \in F$ only, because the other paramaters are never required.

To finish the EM algorithm, one has to detect the convergence of the regularized likelihood (9) that is not computable as it is. However, when the constant

$$\frac{1}{n} \sum_{i=1}^{n} \sum_{k=1}^{d} \log p_{0k}(x_{ik}|\theta_{0k}) \tag{20}$$

is substracted from (9), the following computable quantity is obtained

$$\frac{1}{n} \sum_{i=1}^{n} \log \sum_{\ell=1}^{c} \alpha_\ell \prod_{k \in F} \frac{p_{\ell k}(x_{ik}|\theta_{\ell k})}{p_{0k}(x_{ik}|\theta_{0k})} - \lambda \sum_{\ell=1}^{c} \sum_{k \in F} |\theta_{\ell k} - \theta_{0k}|. \tag{21}$$

Though the likelihood itself cannot be observed, the convergence of (21) implies that of (9).

In summary, our learning algorithm is described as Algorithm 1. An important point is that the active patterns F can be determined by the mining algorithm before actually obtaining $\theta_{\ell k}$ and θ_{0k}. As the regularization constant λ gets larger, the search tree gets smaller and the algorithm becomes more efficient. However, over-regularization may result in meaningless clusters and make the algorithm prone to local minima.

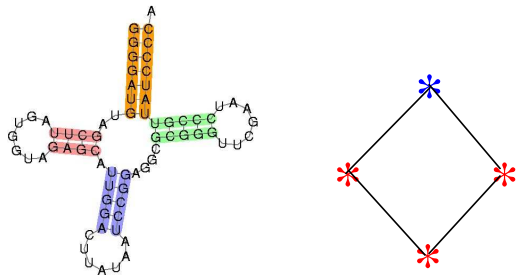

Figure 4. (Left) Example of RNA secondary structure diagram. (Right) The corresponding RNA graph. Node labels are indicated by color.

4. Experiments

To test our algorithm, RNA graphs (Gan et al., 2003) are adopted as the dataset. An RNA is a single-stranded chain of four kinds of nucleotides (A,C,G,U), which takes a complicated shape via hybridization of A-U and C-G pairs (optionally G-U). The structure of an RNA is often represented as a secondary structure diagram (Figure 4, left). A successive chain of hybridized pairs is called a *stem*. For example, stems are highlighted in Figure 4, left. The aim of RNA graphs is to represent topological relationships of stems, not individual nucleotides. As shown in Figure 4, right, one stem corresponds to a node and two nodes are connected by an edge if the stems are linked by an intermediate chain of nucleotides. A node can have a self-loop edge, but, due to the restriction of our graph mining algorithm, the self-loop is encoded as a vertex label. Namely, the node is labeled as 1 if it has a self-loop, and 0 otherwise. It is possible to assign other labels (e.g. stem length) as done by Karklin et al. (2005), but we kept the simple representation to focus on the topological aspects of RNAs.

From numerous families of RNAs registered in Rfam (Griffiths-Jones et al., 2005), we chose three families with completely different functions, namely *Intron GP I* (Int), *SSU rRNA 5* (SSU), and *RNaseP bact a* (RNase). Those families contain long RNAs with a relatively large number of stems, and the size of RNA graphs is comparable among families. We used the first 30, 50 and 50 seed sequences respectively from Int, SSU and RNase.[1] The secondary structure of each RNA is derived using the software *RNAfold* (Hofacker et al., 1994). The common secondary strcuture is imposed by `-C` option, but still the obtained RNA graphs are quite variable (see Figure 5).

[1]Int had only 30 sequences in total.

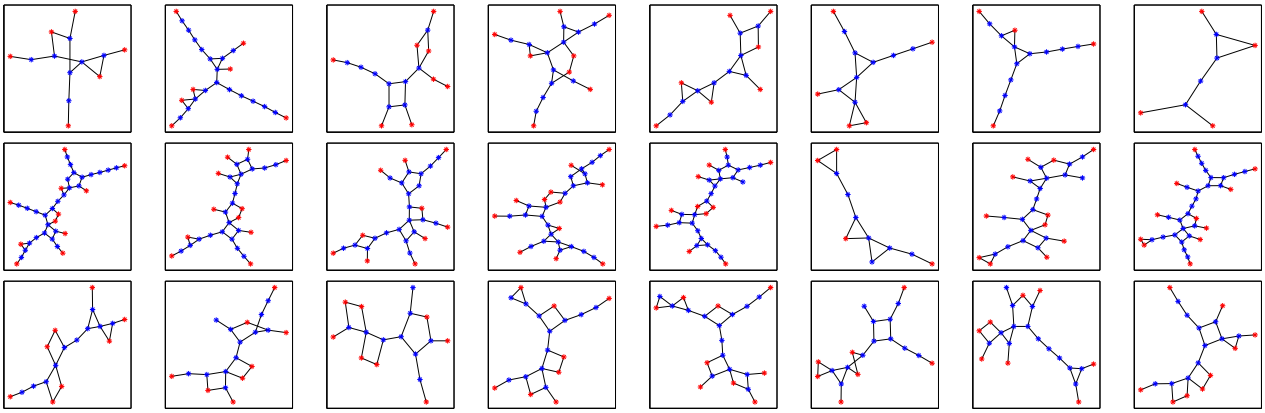

Figure 5. Examples of RNA graphs from Intron-GP-I (top row), SSU-rRNA5 (middle row) and RNaseP-bact-a (bottom row).

Table 1. ROC scores in clustering.

	Int-SSU	Int-RNase	SSU-RNase
MGK	0.748	0.531	**0.878**
Spec	0.550	0.573	0.848
$\lambda = 0.01$	0.824	0.921	0.863
$\lambda = 0.02$	0.821	0.920	0.862
$\lambda = 0.03$	0.825	**0.948**	0.843
$\lambda = 0.04$	0.832	0.947	0.825
$\lambda = 0.06$	0.831	0.941	0.782
$\lambda = 0.08$	**0.845**	0.941	0.787
$\lambda = 0.10$	0.815	0.927	0.786

4.1. Experimental Settings and Results

Our method was compared with the (kernelized) K-means clustering algorithm combined with the marginalized graph kernel (MGK) by Kashima et al. (2003) and the spectral distance (Spec) by Wilson et al. (2005). The EM algorithm was started from ten initial partitions, and the clustering result with the highest regularized likelihood (9) was taken. Notice that the same initial partitions are used for all methods. Out of the three families, three different bipartition problems are made (i.e., Int-SSU, Int-RNase and SSU-RNase). The resulting partitions are evaluated by the ROC score (i.e., the area under the ROC curve), where the likelihood ratios $\log p(\boldsymbol{x}_i|\boldsymbol{\theta}_1) - \log p(\boldsymbol{x}_i|\boldsymbol{\theta}_2)$ are compared with the true class labels.

The ROC scores against diverse regularization strengths are summarized in Table 1. For the MGK, the initial and transition probabilities are determined uniformly. The termination probability was set as $\{0.01, 0.05, 0.1, 0.15, 0.2\}$, and the best result is shown. The competing methods (MGK and Spec) performed well in SSU-RNase, but in the other two problems, our method marked the highest score. Especially, in Int-RNase, the accuracy difference was striking.

In Table 2, the number of patterns and the computational time of our method is shown. The number of patterns decreases as the regularization constant λ increases, and the computational time gets shorter as well, because the search tree can be pruned earlier. Due to the time-consuming mining algorithm, our method was not faster than the competing methods: The average computation time of MGK in the three problems was 19.3, 12.1 and 29.7 seconds, respectively. For Spec, it was 2.4, 1.3 and 3.0 seconds, respectively. However, an important point is that our method employs the subgraph representation which is considered as more informative than path or spectral representations (Gärtner et al., 2003). Furthermore, the discriminant patterns are obtained as F, which are useful for understanding the obtained clusters. For Int-RNase, the selected patterns are sorted due to the discriminant score

$$\max_{\ell=1,\cdots,c} |\sum_{i=1}^{n} w_{\ell i}(2x_{ik} - 1)| \tag{22}$$

and the top patterns are shown in Figure 6. It is found that rather complex subgraphs having more than 7 nodes are essential for discriminating these two families.

4.2. Discussion

The MGK uses the frequency of label paths to derive the similarity. So, it is considered that the description ability of paths was not sufficient especially in Int-RNase. The performance of MGK deeply depends on the design of vertex and edge labels. For example,

Table 2. Number of selected patterns (total computation time in seconds).

	Int-SSU	Int-RNase	SSU-RNase
$\lambda = 0.01$	12505 (71s)	14366 (77s)	17934 (102s)
$\lambda = 0.02$	12596 (75s)	10988 (65s)	11025 (76s)
$\lambda = 0.03$	9799 (66s)	7632 (52s)	8875 (73s)
$\lambda = 0.04$	6904 (57s)	5924 (45s)	6925 (67s)
$\lambda = 0.06$	5093 (47s)	4305 (37s)	5230 (58s)
$\lambda = 0.08$	4065 (42s)	3001 (32s)	3896 (50s)
$\lambda = 0.10$	3245 (37s)	2074 (26s)	2923 (44s)

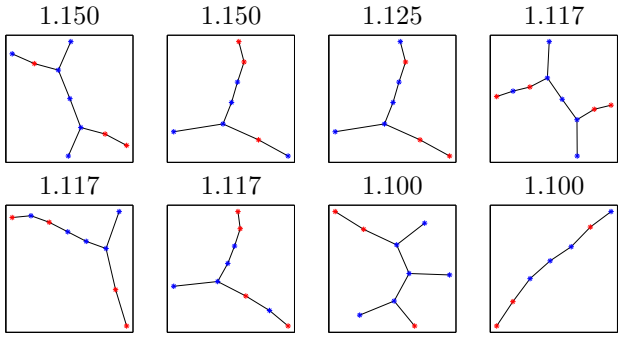

Figure 6. Most discriminative patterns for Int-RNase ($\lambda = 0.03$). The score (22) is shown above each pattern.

if there are no labels at all, the kernel is always one (after normalization), so clustering is impossible. In our setting, the nodes are labeled only with 0 or 1, but the MGK would work better if more informative labels were employed. Due to subgraph patterns, our method can work without any labels, which is a strong point, but, on the other hand, our method cannot take into account the similarity of vertex and edge labels. Therefore, it is difficult to use real-valued labels in our method, whereas the MGK can easily do it by means of e.g., RBF kernels. To take into account the similarity of labels in our method, one possible extension is to use an advanced graph mining method with the taxonomy of labels (Inokuchi, 2004), but it is not tried yet.

The Spec method basically compares the spectrum of two Laplacian matrices, which mainly reflect global topology about graphs. On the other hand, the patterns of our method can take local topology into account. Since our data contains the graphs of substantially different sizes in a cluster, it was difficult for the Spec method to capture the difference of graphs.

5. Conclusion

In this paper, we have presented a novel approach to combine probabilistic inference and graph mining. Despite a simple mixture model is used here, it is possible to extend our learning method for more advanced probabilistic models. For example, our mixture model can be used for semi-supervised learning, when the class labels of a few graphs are known in advance.

The key idea of our algorithm is to use the ℓ_1 regularizer to reduce the number of effective parameters, and carry out the EM algorithm without taking the sum over all patterns. Naturally, our idea can be applied to any subclass of graphs. If tree mining is employed instead of graph mining, the computational time will be much shorter, and tree clustering algorithms are still useful for, e.g., semi-structured texts and phylogenetic trees. One point yet to improve is that the number of selected patterns might be too many for interpretation. For better interpretability, the number of patterns can be reduced by, e.g., selecting closed patterns only (Yan & Han, 2003).

Acknowledgments

Part of this work has been done while KT was at AIST Computational Biology Research Center. KT would like to thank H. Saigo, M. Hamada and H. Kashima for fruitful discussions.

A. Proof of Theorem 1

Let us rewrite the problem as

$$\begin{aligned} \underset{\theta_{\ell k},\xi^+,\xi^-}{\text{argmin}} \quad & -\eta_{\ell k}\theta_{\ell k} + \log(1+\exp(\theta_{\ell k})) + \lambda_\ell(\xi^+ + \xi^-) \\ & \theta_{\ell k} - \theta_{0k} \le \xi^+, \quad \theta_{\ell k} - \theta_{0k} \ge -\xi^- \\ & \xi^+ \ge 0, \quad \xi^- \ge 0. \end{aligned}$$

Introducing the coefficients $\alpha^+, \alpha^-, \delta^+, \delta^- \ge 0$, the Lagrangian is written as

$$\begin{aligned} L = \quad & -\eta_{\ell k}\theta_{\ell k} + \log(1+\exp(\theta_{\ell k})) + \lambda_\ell(\xi^+ + \xi^-) \\ & +\alpha^+(\theta_{\ell k} - \theta_{0k} - \xi^+) - \alpha^-(\theta_{\ell k} - \theta_{0k} + \xi^-) \\ & -\delta^+\xi^+ - \delta^-\xi^-. \end{aligned}$$

Solving L for ξ^+ and ξ^-, the following equations are obtained,

$$\begin{aligned} \tfrac{\partial L}{\partial \xi^+} &= \lambda_\ell - \alpha^+ - \delta^+ = 0 \\ \tfrac{\partial L}{\partial \xi^-} &= \lambda_\ell - \alpha^- - \delta^- = 0. \end{aligned} \tag{23}$$

Since $\delta^+, \delta^- \ge 0$, the inequalities $\alpha^+ \le \lambda_\ell$ and $\alpha^- \le \lambda_\ell$ are induced from (23). Using (23), the Lagrangian is simplified as

$$L = -\eta_{\ell k}\theta_{\ell k} + \log(1+\exp(\theta_{\ell k})) + (\alpha^+ - \alpha^-)(\theta_{\ell k} - \theta_{0k}).$$

Solving it for $\theta_{\ell k}$, we get

$$\frac{\partial L}{\partial \theta_{\ell k}} = -\eta_{\ell k} + \frac{\exp(\theta_{\ell k})}{1+\exp(\theta_{\ell k})} + \alpha^+ - \alpha^- = 0.$$

If $\alpha = \alpha^+ - \alpha^-$, this equation is solved as

$$\theta_{\ell k} = \log(\eta_{\ell k} - \alpha) - \log(1 - \eta_{\ell k} + \alpha).$$

The dual problem is written as

$$\begin{aligned}\operatorname*{argmin}_{-\lambda_\ell \le \alpha \le \lambda_\ell} \quad & (1-\eta_{\ell k}+\alpha)\log(1-\eta_{\ell k}+\alpha) \\ & +(\eta_{\ell k}-\alpha)\log(\eta_{\ell k}-\alpha) + \alpha\theta_{0k}.\end{aligned}$$

Ignoring the constraint, this problem is solved as

$$\alpha = \eta_{\ell k} - \eta_{0k},$$

where the corresponding primal solution is $\theta_{\ell k} = \theta_{0k}$. If $\eta_{\ell k} - \eta_{0k} \le -\lambda_\ell$, the optimal solution is $\alpha = -\lambda_\ell$, and the corresponding primal solution is

$$\theta_{\ell k} = \log \frac{\eta_{\ell k} + \lambda_\ell}{1 - (\eta_{\ell k} + \lambda_\ell)}.$$

On the other hand, if $\eta_{\ell k} - \eta_{0k} \ge \lambda_\ell$, we get $\alpha = \lambda_\ell$ and

$$\theta_{\ell k} = \log \frac{\eta_{\ell k} - \lambda_\ell}{1 - (\eta_{\ell k} - \lambda_\ell)}.$$

References

Gan, H., Pasquali, S., & Schlick, T. (2003). Exploring the repertoire of RNA secondary motifs using graph theory: Implications for RNA design. *Nucleic Acids Res.*, *31*, 2926–2943.

Gärtner, T., Flach, P., & Wrobel, S. (2003). On graph kernels: Hardness results and efficient alternatives. *Proceedings of 16th Annual Conference on Computational Learning Theory and 7th Kernel Workshop (COLT)* (pp. 129–143). Springer Verlag.

Gasteiger, J., & Engel, T. (2003). *Chemoinformatics: a textbook*. Wiley-VCH.

Griffiths-Jones, S., Moxon, S., Marshall, M., Khanna, A., Eddy, S. R., & Bateman, A. (2005). Rfam: annotating non-coding RNAs in complete genomes. *Nucleic Acids Res*, *33*, 121–124.

Hofacker, I. L., Fontana, W., Stadler, P. F., Bonhoeffer, L. S., Tacker, M., & Schuster, P. (1994). Fast Folding and Comparison of RNA Secondary structures. *Monatsh. Chemie*, *125*, 167–188.

Inokuchi, A. (2004). Mining generalized substructures from a set of labeled graphs. *Proceedings of the 4th IEEE International Conference on Data Mining (ICDM)* (pp. 415–418). IEEE Computer Society.

Inokuchi, A., Washio, T., & Motoda, H. (2000). An apriori algorithm for mining frequent substructures from graph data. *European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD)* (pp. 13–23).

Karklin, Y., Meraz, R., & Holbrook, S. (2005). Classification of non-coding RNA using graph representation of secondary structure. *Pac. Symp. Biocomput.*, 4–15.

Kashima, H., Tsuda, K., & Inokuchi, A. (2003). Marginalized kernels between labeled graphs. *Proceedings of the 20th International Conference on Machine Learning* (pp. 321–328). Menlo Park, CA, AAAI Press.

Kudo, T., Maeda, E., & Matsumoto, Y. (2005). An application of boosting to graph classification. In L. Saul, Y. Weiss and L. Bottou (Eds.), *Advances in neural information processing systems 17*, 729–736. Cambridge, MA: MIT Press.

McLachlan, G., & Basford, K. (1998). *Mixture models: Inference and applications to clustering*. Marcel Dekker.

Roth, V., & Lange, T. (2004). Feature selection in clustering problems. In S. Thrun, L. Saul and B. Schölkopf (Eds.), *Advances in neural information processing systems 16*. Cambridge, MA: MIT Press.

Sanfeliu, A., & Fu, K. (1983). A distance measure between attributed relational graphs for pattern recognition. *IEEE Trans. Syst. Man Cybern.*, *13*, 353–362.

Wilson, R., Hancock, E., & Luo, B. (2005). Pattern vectors from algebraic graph theory. *IEEE Trans. Patt. Anal. Mach. Intell.*, *27*, 1112–1124.

Yan, X., & Han, J. (2002). gspan: Graph-based substructure pattern mining. *Proceedings of the 2002 IEEE International Conference on Data Mining (ICDM)* (pp. 721–724). IEEE Computer Society.

Yan, X., & Han, J. (2003). CloseGraph: mining closed frequent graph patterns. *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)* (pp. 286–295). ACM.