PERGAMON

Pattern Recognition 36 (2003) 1311–1324

PATTERN RECOGNITION
THE JOURNAL OF THE PATTERN RECOGNITION SOCIETY

www.elsevier.com/locate/patcog

First order Gaussian graphs for efficient structure classification

Andrew D. Bagdanov*, Marcel Worring[1]

Faculty of Science, University of Amsterdam, Kruislaan 403, Amsterdam 1098 SJ, Netherlands

Received 5 September 2001; accepted 1 August 2002

Abstract

First order random graphs as introduced by Wong are a promising tool for structure-based classification. Their complexity, however, hampers their practical application. We describe an extension to first order random graphs which uses continuous Gaussian distributions to model the densities of all random elements in a random graph. These First Order Gaussian Graphs (FOGGs) are shown to have several nice properties which allow for fast and efficient clustering and classification. Specifically, we show how the entropy of a FOGG may be computed directly from the Gaussian parameters of its random elements. This allows for fast and memoryless computation of the objective function used in the clustering procedure used for learning a graphical model of a class. We give a comparative evaluation between FOGGs and several traditional statistical classifiers. On our example problem, selected from the area of document analysis, our first order Gaussian graph classifier significantly outperforms statistical, feature-based classifiers. The FOGG classifier achieves a classification accuracy of approximately 98%, while the best statistical classifiers only manage approximately 91%.
© 2002 Pattern Recognition Society. Published by Elsevier Science Ltd. All rights reserved.

Keywords: Structural pattern recognition; First order random graphs; Document analysis; Genre classification

1. Introduction

In many pattern classification problems the need for representing the structure of patterns within a class arises. Applications for which this is particularly true include character recognition [1,2], occluded face recognition [3], and document type classification [4–6]. These problems are not easily modeled using feature-based statistical classifiers. This is due to the fact that each pattern must be represented by a single, fixed-length feature vector, which fails to capture its inherent structure. In fact, most local structure information is lost and patterns with the same global features, but different structure, cannot be distinguished.

Structural pattern recognition attempts to address this problem by describing patterns using grammars, knowledge bases, graphs, or other structural models [7,8]. Such techniques typically use rigid models of structure within pattern instances to model each class.

A technique that uses structural models, while allowing statistical variation within the structure of a model, was introduced by Wong [1]. He proposes a random graph model in which vertices and edges are associated with discrete random variables taking values over the attribute domain of the graph. The use of discrete densities complicates the learning and classification processes for random graphs. Graph matching is a common tool in structural pattern recognition [9], but any matching procedure for random graphs must take statistical variability into account. Entropy, or the increment in entropy caused by combining two random graphs, is typically used as a distance metric. When computing the entropy of a random graph based on discrete densities, it is necessary to remember all pattern graphs used to train it. Also, some problems do not lend themselves to discrete modeling, such as when there is a limited amount of training samples, or it is desirable to learn a model from a minimal amount of training data.

* Corresponding author. Tel.: +31-20-525-7518; fax: +31-20-525-7490.
E-mail address: andrew@science.uva.nl (A.D. Bagdanov).

[1] The work was supported in part by the ICES-KIS MIA-project.

In order to alleviate some of the limitations imposed by the use of discrete densities we have developed an extension to Wong's first order random graphs that uses continuous Gaussian distributions to model the variability in random graphs. We call these First Order Gaussian Graphs (FOGGs). The adoption of a parametric model for the densities of each random graph element is shown to greatly improve the efficiency of entropy-based distance calculations. To test the effectiveness of FOGGs as a classification tool we have applied them to a problem from the document analysis field, where structure is the key factor in making distinctions between document classes [10].

The rest of the paper is organized as follows. The next section introduces first order Gaussian graphs. Section 3 describes the clustering procedure used to learn a graphical model from a set of training samples. Section 4 details a series of experiments probing the effectiveness of FOGGs as a classifier for a problem from document image analysis. Finally, a discussion of our results and indications of future directions are given in Section 5.

2. Definitions and basic concepts

In this section we introduce FOGGs. First we describe how individual pattern instances are represented, and then how FOGGs can be used to model a set of such instances.

2.1. First order Gaussian graphs

A structural pattern in the recognition task consists of a set of primitive components and their structural relations. Patterns are modeled using attributed relational graphs (ARGs). An ARG is defined as follows:

Definition 1. An *attributed relational graph*, G, over $L = (L_v, L_e)$ is a 4-tuple (V_G, E_G, m_v, m_e), where V is a set of vertices, $E \subseteq V \times V$ is a set of edges, $m_v : V \longrightarrow L_v$ is the vertex interpretation function, and $m_e : E \longrightarrow L_e$ is the edge interpretation function.

In the above definition L_v and L_e are known respectively as the *vertex attribute domain* and *edge attribute domain*. An ARG defined over suitable attribute domains can be used to describe the observed attributes of primitive components of a complex object, as well as attributed structural relationships between these primitives.

To represent a class of patterns we use a random graph. A random graph is essentially identical to an ARG, except that the vertex and edge interpretation functions do not take determined values, but vary randomly over the vertex and edge attribute domain according to some estimated density.

Definition 2. A *random attributed relational graph*, R, over $L = (L_v, L_e)$ is a 4-tuple (V_R, E_R, μ_v, μ_e), where V is a set of vertices, $E \subseteq V \times V$ is a set of edges, $\mu_v : V \longrightarrow \Pi$ is the vertex interpretation function, and $\mu_e : E \longrightarrow \Theta$ is the edge interpretation function. $\Pi = \{\pi_i \mid i \in \{1, \dots, |V_R|\}\}$ and $\Theta = \{\theta_{ij} \mid i, j \in \{1, \dots, |V_R|\}\}$ are sets of random variables taking values in L_v and L_e respectively.

An ARG obtained from a random graph by instantiating all vertices and edges is called an *outcome graph*. The joint probability distribution of all random elements induces a probability measure over the space of all outcome graphs. Estimation of this joint probability density, however, becomes quickly unpleasant for even moderately sized graphs, and we introduce the following simplifying assumptions:

(1) The random vertices π_i are mutually independent.
(2) A random edge θ_{ij} is independent of all random vertices other than its endpoints v_i and v_j.
(3) Given values for each random vertex, the random edges θ_{ij} are mutually independent.

Throughout the rest of the paper we will use R to represent an arbitrary random graph, and G to represent an arbitrary ARG. To compute the probability that G is generated by R requires us to establish a common vertex labeling between the vertices of the two graphs. For the moment we assume that there is an arbitrary isomorphism, ϕ, from R into G serving to "orient" the random graph to the ARG whose probability of outcome we wish to compute. This isomorphism establishes a common labeling between the nodes in G and R, and consequently between the edges of the two graphs as well. Later we will address how to determine this isomorphism separately for training and classification.

Up to this point, our development is identical to that of Wong [1]. In the original presentation, and in subsequent work based on random graph classification [2,3], discrete probability distributions were used to model all random vertices and edges. For many classification problems, however, it may be difficult or unclear how to discretize continuous features. Outliers may also unpredictably skew the range of the resulting discrete distributions if the feature space is not carefully discretized. Furthermore, if the feature space is sparsely sampled for training the resulting discrete distributions may be highly unstable without resorting to histogram smoothing to blur the chosen bin boundaries. In such cases it is preferable to use a continuous, parametric model for learning the required densities. For large feature spaces, the adoption of a parametric model may yield considerable performance gains as well.

To address this need we will use continuous random variables to model the random elements in our random graph model. We assume that each $\pi_i \sim N(\boldsymbol{\mu}_{\pi_i}, \boldsymbol{\Sigma}_{\pi_i})$, and that the joint density of each random edge and its endpoints is Gaussian as well. We call random graphs satisfying these conditions, in addition to the three first order conditions mentioned earlier, *First Order Gaussian Graphs*, or FOGGs.

Given an ARG $G = (V_G, E_G, m_v, m_e)$ and a FOGG $R = (V_R, E_R, \mu_v, \mu_e)$, the task is now to compute the probability that G is an outcome graph of R. To simplify our notation we let p_{v_i} denote the probability density function of $\mu_v(v_i)$ and $p_{e_{ij}}$ the density of $\mu_e(e_{ij})$. Furthermore, let $\mathbf{v}_i = m_v(\phi(v_i))$ and $\mathbf{e}_{ij} = m_e(\phi(e_{ij}))$ denote the observed attributes for vertex v_i and edge e_{ij} respectively under isomorphism ϕ.

We define the probability that R generates G in terms of a vertex factor

$$V_R(G,\phi) = \prod_{v_i \in V_R} p_{v_i}(\mathbf{v}_i) \tag{1}$$

and an *edge factor*

$$E_R(G,\phi) = \prod_{e_{ij} \in E_R} p_{e_{ij}}(\mathbf{e}_{ij}|\mathbf{v}_i,\mathbf{v}_j). \tag{2}$$

The probability that G is an outcome graph of R is then given by

$$P_R(G,\phi) = V_R(G,\phi) \times E_R(G,\phi). \tag{3}$$

Applying Bayes rule, we rewrite Eq. (2) as

$$E_R(G,\phi) = \prod_{e_{ij} \in E_R} \frac{p_{e_{ij}}(\mathbf{v}_i,\mathbf{v}_j|\mathbf{e}_{ij})\,p_{e_{ij}}(\mathbf{e}_{ij})}{p_{v_i}(\mathbf{v}_i)\,p_{v_j}(\mathbf{v}_j)}, \tag{4}$$

where we may write the denominator as the product of the two vertex probabilities due to the first order independence assumption. Letting $\delta_R(v_i)$ denote the degree of vertex v_i, we can rewrite Eq. (4) as

$$E_R(G,\phi) = \frac{\prod_{e_{ij} \in E_R} p_{e_{ij}}(\mathbf{v}_i,\mathbf{v}_j|\mathbf{e}_{ij})\,p_{e_{ij}}(\mathbf{e}_{ij})}{\prod_{v_i \in V_R} p_{v_i}(\mathbf{v}_i)^{\delta_R(v_i)}}. \tag{5}$$

After substituting Eqs. (1) and (5) into (3) and noting that the vertex probabilities in Eq. (1) cancel with the denominator of Eq. (5) we have

$$P_R(G,\phi) = \frac{\prod_{e_{ij} \in E_R} p_{e_{ij}}(\mathbf{v}_i,\mathbf{v}_j|\mathbf{e}_{ij})\,p_{e_{ij}}(\mathbf{e}_{ij})}{\prod_{v_i \in V_R} p_{v_i}(\mathbf{v}_i)^{\delta_R(v_i)-1}} \tag{6}$$

and by substituting the joint density for the conditional above:

$$\begin{aligned} P_R(G,\phi) &= \frac{\prod_{e_{ij} \in E_R} 1/(p_{e_{ij}}(\mathbf{e}_{ij}))\,p_{e_{ij}}(\mathbf{v}_i,\mathbf{v}_j,\mathbf{e}_{ij})\,p_{e_{ij}}(\mathbf{e}_{ij})}{\prod_{v_i \in V_R} p_{v_i}(\mathbf{v}_i)^{\delta_R(v_i)-1}} \\ &= \frac{\prod_{e_{ij} \in E_R} p_{e_{ij}}(\mathbf{v}_i,\mathbf{v}_j,\mathbf{e}_{ij})}{\prod_{v_i \in V_R} p_{v_i}(\mathbf{v}_i)^{\delta_R(v_i)-1}}. \end{aligned} \tag{7}$$

Recalling that we assume each random vertex is Gaussian we write

$$p_{v_i}(\mathbf{v}_i) = \frac{1}{(2\pi)^{d_1/2}|\mathbf{\Sigma}_{v_i}|^{1/2}} \mathrm{e}^{-1/2(\mathbf{v}_i - \mathbf{\mu}_{v_i})\mathbf{\Sigma}_{v_i}^{-1}(\mathbf{v}_i - \mathbf{\mu}_{v_i})^t}. \tag{8}$$

Letting $p_{w_{ij}}$ denote the (Gaussian) joint probability of edge e_{ij} and its endpoints v_i and v_j, and denoting the concatenation of feature vectors $\mathbf{v}_i$, $\mathbf{v}_j$, and $\mathbf{e}_{ij}$ with $\mathbf{x}_{ij}$ we have

$$p_{w_{ij}}(\mathbf{x}_{ij}) = \frac{1}{(2\pi)^{d_2/2}|\mathbf{\Sigma}_{w_{ij}}|^{1/2}} \mathrm{e}^{-1/2(\mathbf{x}_{ij} - \mathbf{\mu}_{w_{ij}})\mathbf{\Sigma}_{w_{ij}}^{-1}(\mathbf{x}_{ij} - \mathbf{\mu}_{w_{ij}})^t}. \tag{9}$$

Substituting these into (6) and taking the log we arrive at

$$\begin{aligned} \ln P_R(G,\phi) = &\sum_{v_i \in V_R} (\delta(v_i) - 1) \left[\frac{1}{2}(\mathbf{v}_i - \mathbf{\mu}_{v_i})\mathbf{\Sigma}_{v_i}^{-1}(\mathbf{v}_i - \mathbf{\mu}_{v_i})^t \right. \\ &\left. + \ln(2\pi)^{d_1/2}|\mathbf{\Sigma}_{v_i}|^{1/2} \right] \\ &- \sum_{e_{ij} \in E_R} \left[\frac{1}{2}(\mathbf{x}_{ij} - \mathbf{\mu}_{w_{ij}})\mathbf{\Sigma}_{w_{ij}}^{-1}(\mathbf{x}_{ij} - \mathbf{\mu}_{w_{ij}})^t \right. \\ &\left. + \ln(2\pi)^{d_2/2}|\mathbf{\Sigma}_{v_i}|^{1/2} \right]. \end{aligned} \tag{10}$$

This probability and corresponding log-likelihood are central to the use of FOGGs as classifiers. Note that we can promote an ARG to a FOGG by replacing each deterministic vertex and edge with a Gaussian centered on its measured attribute. The covariance matrix for each new random element is selected to satisfy some minimum criterion along the diagonal and can be determined heuristically based on the given problem. Fig. 1 conceptually illustrates the probability metric induced by Eq. (10) over the space of outcome graphs.

2.2. Technicalities

Before continuing with our development of the clustering and classification procedures for FOGGs, it is necessary to first address a few details that will simplify the following development. These details center primarily around the need to compare and combine two FOGGs during the clustering process.

2.2.1. Null extension

During clustering and classification the need will eventually arise to compare, and possibly combine, two FOGGs of different order. Let $R_1 = (V^1, E^1, \mu_v^1, \mu_e^1)$ and $R_2 = (V^2, E^2, \mu_v^2, \mu_e^2)$ be two FOGGs with $n = |V^1|$ and $m = |V^2|$. Furthermore, let $V^1 = \{v_1, \ldots, v_n\}$ and $V^2 = \{u_1, \ldots, u_m\}$. Assume without loss of generality that $m < n$.

We will use the same technique as Wong [1] to extend R_2 by adding null vertices to V^2. Thus we redefine V^2 as

$$V^2 = V^2 \cup \{u_{m+1}, \ldots, u_n\},$$

where the $u_{m+1}, \ldots, u_n$ are *null vertices*, i.e. they have no attribute values, but rather act as place holders so that R_1 and R_2 are of the same order.

Once R_2 has been extended in this fashion, both R_1 and R_2 may be extended to complete graphs through a similar addition of null edges to each graph until edges exist between

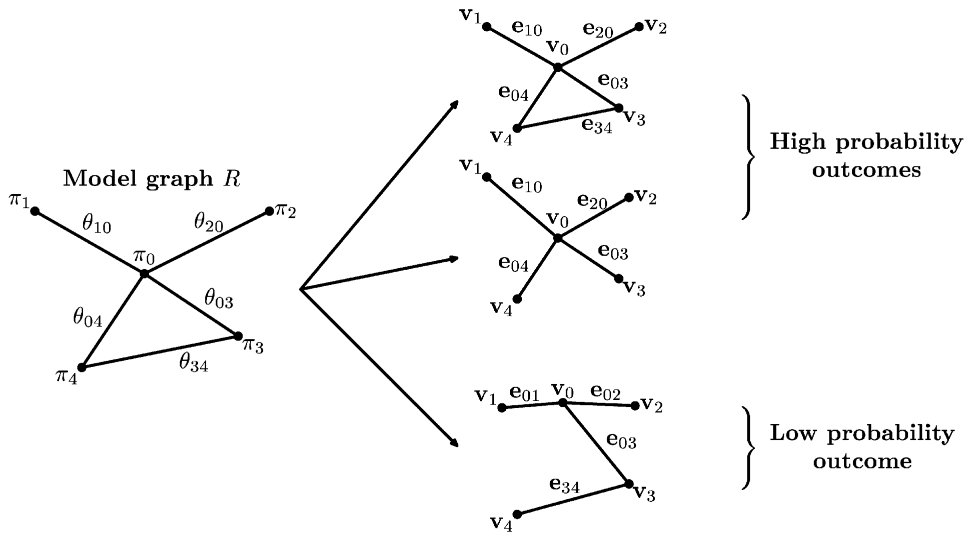


Fig. 1. The probability metric induced by $P_R(G, \phi)$ over the space of outcome graphs. The probability depends not only on structural variations, but deviation from the expected value of each corresponding random variable π_i and θ_i. This is illustrated spatially, however the vertex and edge attribute domains need not have a spatial interpretation.

all vertices. By adding these null graph elements we can now treat R_1 and R_2 as being structurally isomorphic, so that we are guaranteed that an isomorphism exists and we must only search for an optimal one.

Our probabilistic model must also be enriched to account for such structural modifications to random graphs. First, note that our densities $p_{v_i}(\mathbf{x})$ modeling the features of each random vertex are actually conditional probabilities:

$$p_{v_i}(\mathbf{x}) = p_{v_i}(\mathbf{x} | \phi(v_i) \text{ is non-null}).$$

After all, we can only update the feature distribution of a vertex, or indeed even evaluate it, when we have new information about an actual non-null outcome. To account for the possibility of a vertex or edge not being instantiated, we will additionally keep track of a priori probabilities of a random element generating a non-null outcome:

$$p_R(v_i) = \text{probability } v_i \in V_R \text{ is non-null},$$

$$p_R(e_{ij}) = \text{probability } e_{ij} \in E_R \text{ is non-null}. \tag{11}$$

Thus, whenever we wish to evaluate the probability of a random element λ taking the value $\mathbf{x}$ we will use $p(\lambda)p_\lambda(\mathbf{x})$, which is intuitively the probability that λ exists *and* takes the value $\mathbf{x}$. Whenever a probability for a random vertex or edge must be evaluated on a null value, we will fall back to the prior probability of that element. This is done by optimistically assuming that the null element results from a detection failure, and that the missing feature is the expected value of the random element it is being matched with.

Through the use of such extensions, we can compare graphs with different sized vertex sets. For the remainder of the paper we will assume that such an extension has been performed, and that any two graphs under consideration are of the same order.

2.2.2. *Entropy of FOGGs*

In order to measure the quality of a model for a class we require some quantitative measure that characterizes the outcome variability of a FOGG. As variability is statistically modeled, Shannon's entropy is well suited for this purpose [11].

We can write the entropy in a FOGG as the sum of the contributions of the vertices and edges:

$$H(R) = \{H(V_R) + H(E_R)\}. \tag{12}$$

Because of the first order assumptions of independence, we can write the vertex and edge entropies as the sum of the entropy in each component. The entropy in each random vertex and edge may be written as the sum of the entropy contributed by the feature and prior entropy:

$$H(v_i) = H(\mu_v(v_i)) + H(p(v_i)),$$

$$H(e_{ij}) = H(\mu_e(e_{ij})) + H(p(e_{ij})). \tag{13}$$

Eq. (12) then becomes

$$H(R) = \sum_{v_i \in V_R} H(v_i) + \sum_{e_{ij} \in E_R} H(e_{ij}). \tag{14}$$

For clustering we are primarily interested in the increment of entropy caused by conjoining two graphs. We denote by $R_{1\oplus 2}(\phi)$ the graph resulting from conjoining the Gaussian random graphs R_1 and R_2 according to the isomorphism ϕ between R_1 and R_2. Assuming without loss of generality that

$H(R_1) \leqslant H(R_2)$, we can write the increment in entropy as

$$\Delta H(R_1,R_2,\phi) = H(R_{1\oplus 2}(\phi)) - H(R_1)$$

and then substitute the sum of the component entropies:

$$\Delta H(R_1,R_2,\phi) = \sum_{v_i \in V_{R_{1\oplus 2}(\phi)}} H(\Delta\mu_v(v_i)) - \sum_{v_i \in V_{R_1}} H(\mu_v^1(v_i)) + \sum_{e_{ij} \in E_{R_{1\oplus 2}(\phi)}} H(\Delta\mu_e(e_{ij})) - \sum_{e_{ij}\in E_{R_1}} H(\mu_e^1(e_{ij})). \tag{15}$$

We use $\Delta\mu_\lambda(x)$ to denote the density of the random variable λ updated to reflect the observations of the corresponding random variable as dictated by the isomorphism ϕ.

From Eqs. (13) and (14) we can express the increment in entropy as the sum of the increment in the feature density and the prior distribution. Since we are using Gaussian distributions to model each random element, the entropy of a Gaussian:

$$H(X) = \ln(2\pi e)^{N/2} |\mathbf{\Sigma}_X|^{1/2} \tag{16}$$

for Gaussian random variable X will prove useful as this will allow us to compute component entropies directly from the parameters of the corresponding densities. The technique for estimating the parameters of the combined distribution is described next.

2.2.3. Parameter estimation

Given two Gaussian random variables X and Y, and samples $\{x_1,\ldots,x_n\}$, $\{y_1,\ldots,y_n\}$ from each distribution, we estimate the Gaussian parameters in the normal fashion

$$\bar{\mathbf{x}} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{x}_i, \quad \mathbf{\Sigma}_X = \frac{1}{n-1}\sum_{i=1}^{n} \mathbf{x}_i\mathbf{x}_i^t - n\bar{\mathbf{x}}\bar{\mathbf{x}}^t,$$

$$\bar{\mathbf{y}} = \frac{1}{m}\sum_{i=1}^{m} \mathbf{y}_i, \quad \mathbf{\Sigma}_Y = \frac{1}{m-1}\sum_{i=1}^{m} \mathbf{y}_i\mathbf{y}_i^t - m\bar{\mathbf{y}}\bar{\mathbf{y}}^t. \tag{17}$$

Assuming that the samples from X and Y are generated by a single Gaussian Z, we can compute the Gaussian parameters for Z directly from the estimates of X and Y:

$$\bar{\mathbf{z}} = \frac{1}{n+m}(n\bar{\mathbf{x}} + m\bar{\mathbf{y}}),$$

$$\mathbf{\Sigma}_Z = \frac{1}{m+n-1}\left(\sum_{i=1}^{n}\mathbf{x}_i\mathbf{x}_i^t + \sum_{i=1}^{m}\mathbf{y}_i\mathbf{y}_i^t - (m+n)\bar{\mathbf{z}}\bar{\mathbf{z}}^t\right)$$

$$= (n-1)\mathbf{\Sigma}_X + n\bar{\mathbf{x}}\bar{\mathbf{x}}^t + (m-1)\mathbf{\Sigma}_Y + m\bar{\mathbf{y}}\bar{\mathbf{y}}^t - (m+n)\bar{\mathbf{z}}\bar{\mathbf{z}}^t. \tag{18}$$

Eq. (18) gives us a fast method for computing the entropy arising from combining two random vertices. It also allows us to compute the parameters of the new distribution without having to remember the samples that were used to estimate the original parameters. When there are too few observations to robustly estimate the covariance matrices, $\mathbf{\Sigma}_X$ is chosen to reflect the inherent uncertainty in a single (or very few) observation(s). This also allows us to promote an ARG to a FOGG by setting each mean to the observed feature, and setting the covariance matrices to this minimal $\mathbf{\Sigma}$.

2.3. Discussion

At this point it is useful to take a step back from the mathematical technicalities presented in the previous subsection and examine the practical importance they represent. By replacing the original discrete random variables with continuous ones, we have eliminated the need to discretize our feature space. This, in conjunction with the adoption of a Gaussian model for each random variable, additionally minimizes the complexity of updating the estimated distribution and entropy of a random element.

Consider the process of conjoining two discrete distributions. In the worst case, every bin in the resulting distribution must be updated. The complexity of this procedure will be proportional to the size of the quantized feature space. Computing the increase in entropy caused by joining two discrete distributions will have the same complexity. Using Gaussian distributions, however, Eqs. (17) and (18) allow us to compute the parameters of a new distribution, and Eq. (16) to compute the increment in entropy directly from the parameters of the new distribution. This reduces the complexity to d^2, where d is the dimensionality of the feature space.

3. Clustering and classification

In this section we describe the technique for synthesizing a FOGG to represent a set of input pattern ARGs. The approach uses hierarchical clustering of the input ARGs, which yields a clustering minimizing the entropy of the resulting FOGG(s). Entropy is useful in that it characterizes the intrinsic variability in the distribution of a FOGG over the space of possible outcome graphs.

3.1. Hierarchical clustering of FOGGs

The concepts of increment in entropy introduced in Section 2.2.2 can now be used to devise a clustering procedure for FOGGs. The first step is to derive a distance measure between FOGGs that is based on the minimum increment in entropy. Using Eq. (15), the *minimum* increment of entropy for the merging of two FOGGs can be written:

$$\Delta H(R_1,R_2) = \min_{\phi}\{\Delta H(R_1,R_2,\phi)\}, \tag{19}$$

where the minimization is taken over all possible isomorphisms ϕ between R_1 and R_2.

At last we have arrived at the need to establish an actual isomorphism between two graphs. Unfortunately this problem is NP-hard, and we must settle for an approximation to the optimal isomorphism. We choose to optimize only over the vertex entropy $H(V_R, \phi)$. This approximation is acceptable for problems where much of the structural information is present in the vertex observations. Edge probabilities are still used in the classification phase, so gross structural deviations will not result in misclassifications.

There are two ways in which the entropy of a vertex may be changed by conjoining it with a vertex in another FOGG. The feature density of the first vertex may be modified to accommodate the observations of the random vertex it is matched with according to ϕ. Or, when ϕ maps v_i to a null vertex, the entropy may be changed due to a decrease in its prior probability p_{v_i} of it being instantiated in an outcome graph.

Using Eq. (16) we may write the increment in vertex entropy due to the feature distribution as

$$\Delta H_f(\mu_v(v_i), R_2, \phi) = \ln(2\pi e)^{N/2}(|\mathbf{\Sigma}_{\Delta\mu(v_i)}|^{1/2} - |\mathbf{\Sigma}_{\mu^1(v_i)}|^{1/2}). \tag{20}$$

Eq. (18) gives us a method for rapidly computing the covariance matrix for each random element in the new graph $R_{1\oplus 2}$, and thus the increment in entropy.

For the increment in prior entropy, we first note that the prior probabilities p_{v_i} for each vertex will be of the form n_i/N_i, where n_i is the number of times vertex v_i was instantiated as a non-null vertex, and N_i is the total number of training ARGs combined thus far in a particular cluster. We can then write the change in prior entropy as

$$\begin{aligned}\Delta H_p(p(v_i), R_2, \phi) &= -[p'(v_i)\ln p'_{v_i} + (1-p'(v_i))\ln(1-p'(v_i))] \\ &\quad - [p(v_i)\ln p_{v_i} + (1-p(v_i))\ln(1-p(v_i))],\end{aligned} \tag{21}$$

where

$$p'(v_i) = \begin{cases} \frac{n_i+1}{N_i+1} & \text{if } \phi(v_i) \text{ is non-null,} \\ \frac{n_i}{N_i+1} & \text{if } \phi(v_i) \text{ is null.} \end{cases} \tag{22}$$

We solve the optimization problem given by Eq. (19) using the maximum weight matching in a bipartite graph. Given two FOGGs R_1 and R_2 of order n, construct the complete bipartite graph $K_{n,n} = \bar{K}_n \times \bar{K}_n$. A matching in $K_{n,n}$ is a subset of edges such that the degree of each node in the resulting graph is exactly one. The maximum weight matching is the matching that maximizes the sum of the edge weights in the matching. By weighting each edge in the bipartite graphs with (see Fig. 2):

$$w_{ij} = -\Delta H(\mu_v(v_i), R_2, \phi)$$

with $\Delta H(\mu_v(v_i), R_2, \phi)$ as given in Eq. (20) and solving for the maximum weight matching, we solve for the isomorphism that minimizes the increment in vertex entropy. There exist efficient, polynomial time algorithms for solving the maximum weight matching problem in bipartite graphs [12]. The complexity of the problem O(n^3) using the Hungarian method, where n is the number of vertices in the bipartite graph.

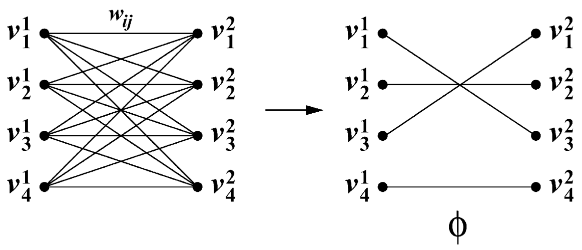


Fig. 2. Computation of the sub-optimal isomorphism ϕ for two first order Gaussian graphs. Each edge is weighted with w_{ij}, whose value is the increment in entropy caused by conjoining the estimated distributions of random vertices v_i^1 and v_j^2. The same technique will be used for determining an isomorphism for classification as well, but with w_{ij} representing the probability that random vertex v_i^1 takes the value v_j^2.

Now we may construct a hierarchical clustering algorithm for FOGGs. For a set of input ARGs, we desire, on the one hand, a minimal set of FOGGs that may be used to model the input ARGs. On the other hand, we also wish to minimize the resulting entropy of each FOGG by preventing unnatural combinations of FOGGs in the merging process.

The algorithm should return a set of FOGGs, $\mathscr{R}_i = \{R_1^i, \dots R_{m_i}^i\}$, that represent the original set of attributed graphs. We will call this set of random graphs the *graphical model* of the class of ARGs. An entropy threshold h controls the amount of variability allowed in a single FOGG. This threshold parameter controls the tradeoff between the number of FOGGs used to represent a class and the amount of variability, i.e. entropy, allowed in any single FOGG in the graphical model. Algorithm 1 provides pseudo-code for the hierarchical synthesis procedure. In the supervised case, the algorithm may be run on each set of samples from the pre-specified classes. For unsupervised clustering, the entire unlabeled set of samples may be clustered. The entropy threshold h may be used to control the number of FOGGs used to represent the class by limiting the maximum entropy allowed in any single FOGG. Fig. 3 graphically illustrates the learning process for FOGGs.

Algorithm 1. Synthesize FOGG(s) from a set of ARGs

```
Input: $\mathscr{G} = \{G_1, \dots, G_n\}$, a set of ARGs, and $h$, a
maximum entropy threshold.
Output: $\mathscr{R} = \{R_1, \dots, R_m\}$, a set of FOGGs representing $\mathscr{G}$.
  Initialize $\mathscr{R} = G$, promoting each ARG to a FOGG
  (Section 2.2).
  Compute $H = [h_{ij}]$, the $n \times n$ distance matrix, with
  $h_{ij} = \Delta H(R_i, R_j)$.
  Let $h_{kl} = \min h_{ij}$.
  while ($|R| > 1$ and $H(R_i) + h_{kl} < h$) do
    Form the new FOGG $R_{k\oplus l}$, add it to $\mathscr{R}$, remove
    $R_k$ and $R_l$ from $\mathscr{R}$.
```

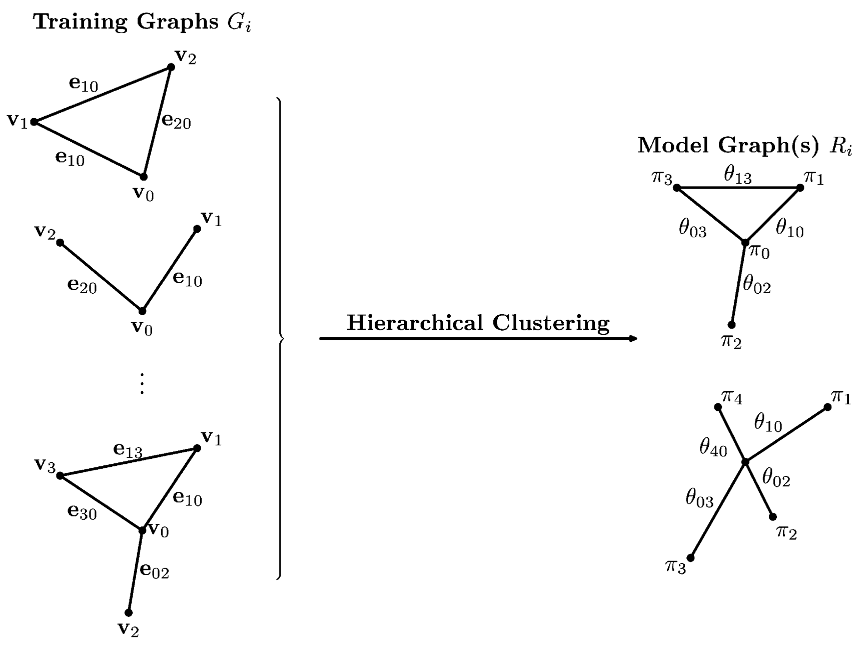


Fig. 3. The learning process for first order Gaussian graphs. A set of sample graphs are synthesized into a set one or more random graphs which represent the class. The hierarchical clustering process described in Algorithm 1 chooses combinations and isomorphisms that minimize the increment in vertex entropy arising from combining two random graphs.

```
    Update distance matrix $H$ to reflect the new and
    deleted FOGGs.
    Re-compute $h_{kl}$, the minimum entropy increment
    pair.
  **end while**
```

Note that this clustering procedure requires, for the creation of the initial distance matrix alone, the computation of $n(n-1)$ isomorphisms, where n is the number of input ARGs. Subsequent updates of the distance matrix will demand a total of $O(n^2)$ additional isomorphism computations in the worst case. Each isomorphism additionally requires the computation of m^2 entropy increments as given in Eq. (20), where m represents the number of vertices in the graphs being compared. Using the techniques derived in Section 2.2 we can exploit the use of Gaussian distributions to greatly improve the efficiency of these computations. This, combined with the use of the bipartite matching approach as an approximation in finding the optimal isomorphism will enhance the overall efficiency of the clustering procedure.

3.2. Classification using FOGGs

Given a set of graphical models for a number of classes, we construct a maximum likelihood estimator as follows. Let $\mathscr{R}_i = \{R_1^i, \ldots, R_{m_i}^i\}$ be the graphical model for class i. Our classifier, presented with an unknown ARG G, should return a class label w, from a set of known classes $\{w_1, \ldots, w_n\}$. The maximum likelihood classifier returns:

$$w_i, \quad \text{where } i = \arg\max_i \left\{ \max_{1 \leqslant j \leqslant m_i} \max_{\phi} P_{R_j^i}(G, \phi) \right\}$$

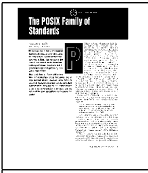
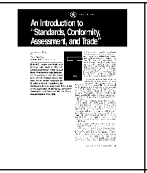
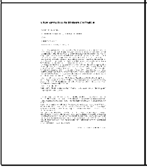
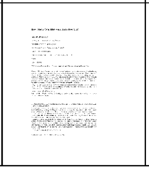
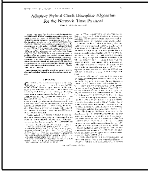
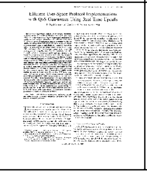
with $P_{R_j^i}(G)$ as defined in Eq. (3). This procedure is described in more detail in Algorithm 2.

Algorithm 2. Classification with first order Gaussian graphs

```
**Input:** $G$, an unclassified ARG,
  $\mathscr{R} = \{\mathscr{R}_1, \ldots, \mathscr{R}_n\}$, graphical models representing pattern
  classes $\omega_1, \ldots, \omega_n$,
  with each $\mathscr{R}_i = \{R_1^i, \ldots, R_{m_i}^i\}$ a set of FOGGs.
**Output:** $\omega_i$ for some $i \in \{1, \ldots, n\}$
  **for all** $\mathscr{R}_i \in \mathscr{R}$ **do**
    Set $P_i = 0$.
    **for all** $R_j^i \in \mathscr{R}_i$ **do**
      Compute orientation $\phi$ of $R_j^i$ w.r.t. $G$ that
      maximizes $P_{R_j^i}(G, \phi)$ (Fig. 2).
      $P_i = \max\{P_{R_j^i}, P_i\}$
    **end for**
  **end for**
  $k = \arg\max_i(P_i)$
  **return** $\omega_k$
```

In establishing the isomorphism ϕ for classification, it is useful to use the log-likelihood function given in Eq. (10). We can then use the same bipartite graph matching technique

Table 1
Sample document images from four of the document genres in the test sample

		IJNM			STDV
		JACM			TNET

introduced in Section 3 and shown in Fig. 2. Instead of weighting each edge with the increment in entropy, however, we weight each edge with the log of the vertex factor from Eq. (8):

$$w_{ij} = -\tfrac{1}{2}(\mathbf{v}_j - \boldsymbol{\mu}_{v_i})\boldsymbol{\Sigma}_{v_i}^{-1}(\mathbf{v}_j - \boldsymbol{\mu}_{v_i})^t - \ln(2\pi)^{d_1/2}|\boldsymbol{\Sigma}_{v_i}|^{1/2}.$$

Determining the maximum weight matching then yields the isomorphism that maximizes the likelihood of the vertex densities.

The entropy threshold h required by the training procedure described in Algorithm 1 has quite an influence over the resulting classifier. Setting $h=0$ would not allow any FOGGs to be combined, resulting in a nearest neighbor classifier. For $h \to \infty$ all classes will be modeled with a single FOGG, with arbitrarily high entropy allowed in an individual FOGG. It is important to select an entropy threshold that balances the tradeoff between the complexity of the resulting classifier and the entropy inherent within each class.

4. Experiments

We have applied the technique of FOGGs to a problem from the document analysis field. In many document analysis systems it is desirable to identify the type, or *genre*, of a document before high-level analysis occurs. In the absence of any textual content, it is essential to classify documents based on visual appearance alone. This section describes a series of experiments we performed to compare the effectiveness of FOGGs with traditional feature-based classifiers.

4.1. Test data

A total of 857 PDF documents were collected from several digital libraries. The sample contains documents from five different journals, which determine the classes in our classification problem. Table 1 gives some example images from four of the genres.

All documents in the sample were converted to images and processed with the ScanSoft TextBridge[2] OCR system, which produces structured output in the XDOC format. Only the layout information from the first page of a document is used since it contains most of the genre-specific information. The importance of classification based on the structure of documents is immediately apparent after a visual inspection of the test collection. Many of the document genres have similar, if not identical, global typographical features such as font sizes, font weight, and amount of text.

4.2. Classifiers

To compare the effectiveness of genre classification by first order random graphs with traditional techniques, a variety of statistical classifiers were evaluated along with the Gaussian graph classifier. The next two subsection detail the specific classifiers studied.

4.2.1. First order Gaussian graph classifier

In this section we develop our technique for representing document layout structure using attributed graphs, which naturally leads to the use of FOGGs as a classifier of document genre. For representing document images, we define the vertex attribute domain to be the vector space of text zone features. A document D_i is described by a set of text zone feature vectors as follows:

$$D_i = \{z_1^i, \ldots, z_{n_i}^i\},$$

where

$$z_j^i = (x_j^i, y_j^i, w_j^i, h_j^i, s_j^i, t_j^i). \qquad (23)$$

In the above definition of a text zone feature vector,

- x_j^i, y_j^i, w_j^i and h_j^i denote the center, width and height of the textzone.

[2] TextBridge is a registered trademark of ScanSoft, inc.

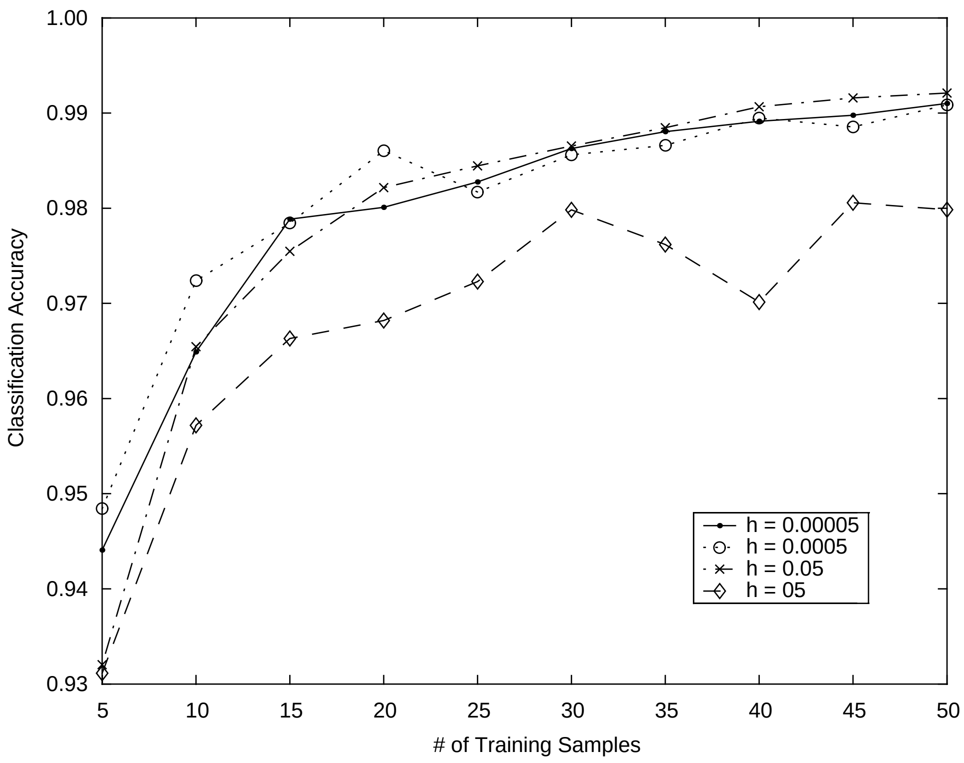


Fig. 4. Classification accuracy over a range of training sizes for various entropy thresholds. The x-axis represents the number of training samples selected randomly *per class*. The y-axis represents the estimated classification accuracy for the corresponding number of training samples.

- s_j^i and t_j^i denote the average pointsize and number of textlines in the zone.

Each vertex in the ARG corresponds to a text zone in the segmented document image. Edges in our ARG representation of document images are *not* attributed. The presence of an edge between two nodes is used to indicate the Voronoi neighbor relation [13]. We use the Voronoi neighbor relation to simplify our structural representation of document layout. We are interested in modeling the relationship between neighboring textzones only, and use the Voronoi neighbor relation to identify the important structural relationships within a document.

Given training samples from a document genre, we construct a graphical model according to Algorithm 1 to represent the genre. The entropy threshold is particularly important for this application. The threshold must be selected to allow variability in document layout arising from minor typographical variations and noisy segmentation, while also allowing for gross structural variations due to common typesetting techniques. For example, one genre may contain both one and two column articles. The threshold should be selected such that the FOGGs representing these distinct layout classes are not combined while clustering.

4.2.2. Statistical classifiers

Four feature-based statistical classifiers were evaluated in comparison with the FOGG classifier. The classifiers considered are the 1-NN, linear-oblique decision tree [14], quadratic discriminant, and linear discriminant classifiers. Global page-level features were extracted from the first page of each document. Each document is represented by a 23 element feature vector as

$$\Bigg(\underbrace{n_p, n_f, n_{iz}, n_{tz}, n_{iz}, n_{tl}}_{\text{global document features}}, \overbrace{p_{ta}, p_i, p_{te}, p_{in}, p_{it}, p_r, p_b}^{\text{proportional zone features}}, \underbrace{h_0, h_1, \ldots, h_9}_{\text{text histogram}}\Bigg).$$

The features are categorized as follows:

- *Global document features*, which represent global attributes of the document. The global features we use are the number of pages, fonts, image zones, text zones, and textlines in the document.
- *Proportional zone features*, that indicate the proportion of document page area classified by the layout segmentation

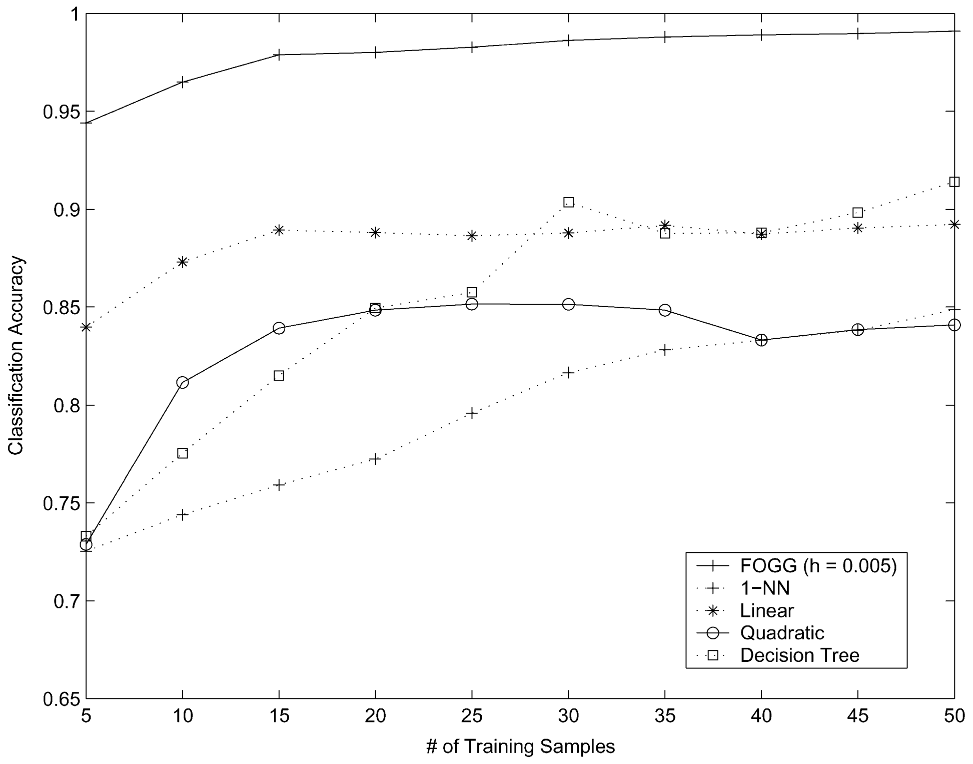


Fig. 5. Learning curves for all classifiers. The curves indicate the estimated classification accuracy of each classifier over a range of training sample sizes. Classification accuracy was estimated by averaging over 50 trials for each sample size. The x-axis represents the number of training samples selected randomly *per class*.

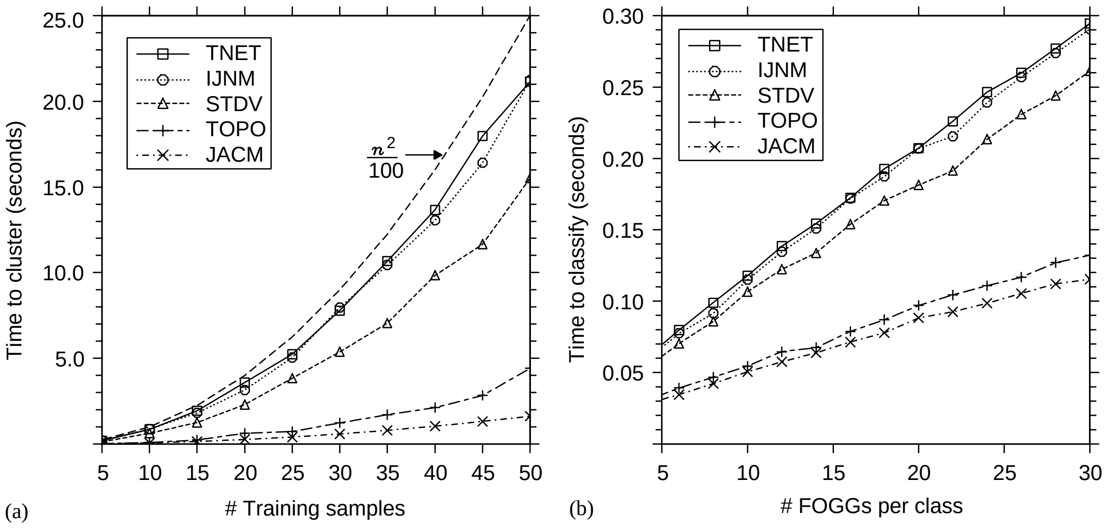


Fig. 6. Computation time for clustering and classification: (a) gives average time required for clustering for each class as a function of the number of training samples per class. A plot of $n^2/100$ is also shown for reference, (b) shows the average time required to classify an unknown ARG as a function of the number of FOGGs representing each class.

Table 2
Average confusion matrix for a linear discriminant classifier. Classification accuracy is estimated over 50 random trials

	IJNM	JACM	STDV	TNET	TOPO
IJNM	0.8524	0.0464	0.0250	0.0750	0.0012
JACM	0	0.9718	0	0	0.0282
STDV	0.0540	0.0016	0.9127	0.0310	0.0008
TNET	0.0199	0.0013	0.0584	0.9195	0.0009
TOPO	0.0724	0.0444	0.0047	0	0.8786

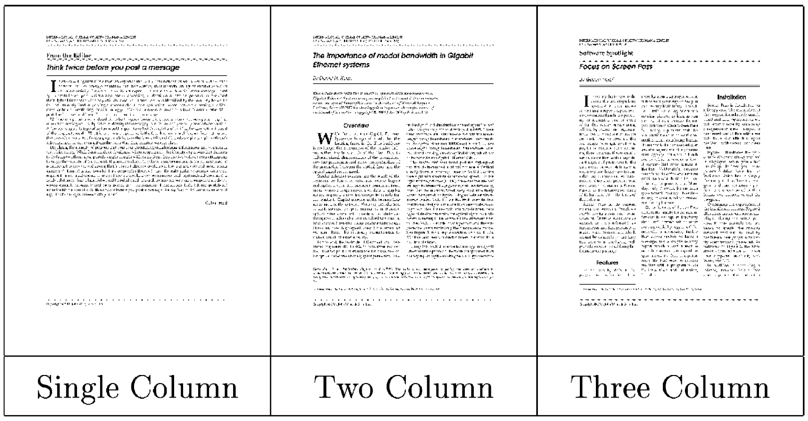


Fig. 7. Example document images from different structural categories within the IJNM class.

process as being a specific type of image or text zone. The feature vector includes the proportion of image area classified as table, image, text, inverse printing, italic, text, roman text, and bold text.

- *Text histogram*, which is a normalized histogram of pointsizes occurring in the document.

This feature space representation is similar to that used by Shin et al. [5], for their experiments in document genre classification. We do not include any texture features from the document image, however. Note that the features for the vertices in the FOGG classifier discussed in the previous subsection is essentially a subset of these features, with a limited set of features collected locally for each text zone rather than for the entire page.

4.3. Experimental results

The first set of experiments we performed was designed to determine the appropriate entropy threshold h for our classification problem and test set. Fig. 4 gives the learning curves for the FOGG classifier over a range of training sample sizes and for several entropy thresholds.

The learning curves indicate that our classifier performs robustly for all but the highest thresholds. This implies that there is intrinsic structural variability in most classes, which cannot be represented by a single FOGG. This is particularly true for small training samples. Note, however, that a relatively high threshold ($h = 0.05$) may be used with no performance loss when sample size is more than 25. This indicates that a smaller and more efficient graphical model may be used to represent each genre if the training sample is large enough.

The second set of experiments provides a comparative evaluation of our classifier with the statistical classifiers described above. Fig. 5 gives the learning curves of all classifiers evaluated. The curves were obtained by averaging the classification results of 50 trials where n samples were randomly selected from each genre for training, and the rest used for testing. These results indicate that the FOGG classifier consistently outperforms all of the other classifiers.

4.4. Computational efficiency

Fig. 6 indicates the computation time required for clustering and classification using FOGGs. The times were estimated by averaging over 50 experimental trials. Fig. 6a shows the time to learn a graphical model of each class as a function of the number of training samples. A plot of $n^2/100$ is also shown for reference. The n^2 bound on clustering time indicates that the time required for clustering is dominated

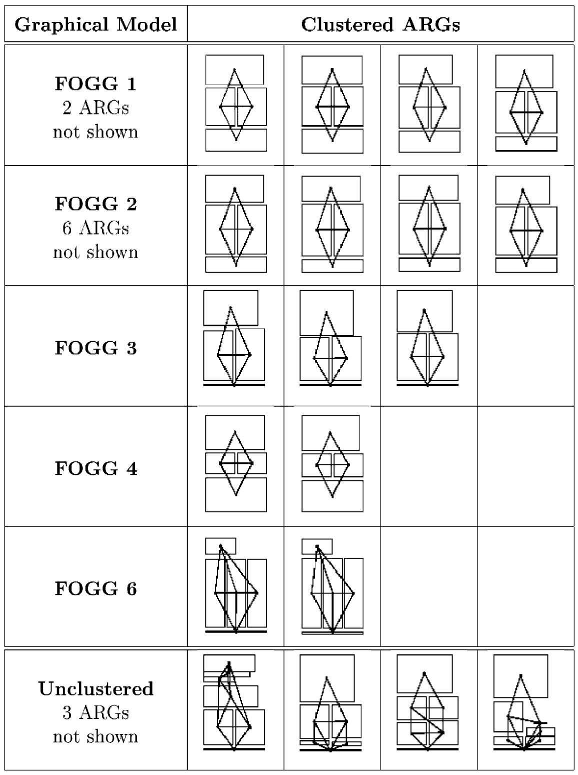


Fig. 8. Example clusters learned from 30 randomly selected samples from the IJNM class of documents. FOGGs 1 and 2 represent the most commonly occurring two-column documents, while FOGG 6 represents the three-column layout style. A very low entropy threshold was used to halt the clustering procedure at this point for illustrative purposes. The unclustered ARGs have not been allowed to merge with any others at this point because they represent segmentation failures and their inclusion in another FOGG would drastically increase the overall entropy.

by the computation of the initial $n \times n$ distance matrix, with an additional constant factor representing the intrinsic complexity of each class. The constant factor of $1/100$ shown in the figure was determined experimentally, but is clearly class specific and depends on the variance in size of the training ARGs being clustered.

Fig. 6b shows the time required to classify an unknown ARG as a function of the number of FOGGs used to represent each class. As expected, there is a linear dependence between the number of FOGGs in a class and the time required for classification. Again, the constant factor affecting the slope of each performance curve is determined by the complexity of each class.

4.5. Analysis

The experimental results presented above indicate that the FOGG classification technique outperforms statistical classifiers on our example application. In understanding the reasons for this it is useful to analyze some specific examples from these experiments. Table 2 gives an average confusion table for a linear discriminant classifier on our dataset.

Note that the class IJNM contains the highest percentage of confusions. The primary reason for this is that the class contains many structural sub-classes within it. Fig. 7 provides examples from the three primary structural sub-classes within the IJNM class. As shown in the figure, the class contains document images with one, two, and three columns of text. The variance in structural layout composition makes it difficult for statistical classifiers to learn decision boundaries distinguishing this class from others. Similar structural sub-classes can also be seen in the TOPO class, which also accounts for the large number of confusions.

The main advantage of the FOGG approach over purely statistical classifiers is in its ability to learn this type of sub-class structure. Fig. 8 gives an example snapshot of the clustering procedure on the IJNM class for a random training sample, with the clustering halted at a low entropy threshold of 0.00005. The figure shows how the distinct structural sub-classes are represented by individual FOGGs. It is this independent modeling of sub-classes that enables the FOGG classifier to outperform statistical methods.

5. Conclusions and future work

We have described an extension to discrete first order random graphs which uses continuous Gaussian distributions for modeling the densities of random elements in graphs. The technique is particularly appealing for its simplicity in learning and representation. This simplicity is reflected in the ability to learn the distributions of random graph elements without having to worry about the discretization of the underlying feature space, and the ability to do so using relatively few training samples. The learned distributions are also effectively "memoryless" in that we do not have to remember the observed samples in order to update density estimates or compute their entropy.

Since we can quickly compute the increment in entropy caused by joining two distributions directly from the Gaussian parameters, the hierarchical clustering algorithm used to learn a class model is very efficient. The use of an approximate matching strategy for selecting an isomorphism to use when comparing random graphs also enhances the efficiency of the technique while preserving discriminatory power.

The experimental results in the previous section establish the effectiveness of first order Gaussian graphs as a classification technique on real-world problems. The need for modeling *structure* is evident, as the statistical classifiers fail to capture the sub-structural composition of some classes in our experiments. While it might be possible to enrich the feature space in such a way that allows statistical classifiers to handle such structural information, the FOGG classifier is already capable of modeling these structural relationships. Moreover, the FOGG classifier uses a small subset of the feature space used by the statistical classifiers in our experiments. The important difference is that the FOGG classifier models the structural relationship between local feature measurements.

The FOGG classifier requires the selection of an appropriate entropy threshold for training. This threshold is problem and data dependent, and was more or less arbitrarily chosen for our experiments. The results shown in Fig. 4 indicate that an adaptive entropy thresholding strategy might be effective for balancing the intrinsic entropy in a class with the desire for minimizing the complexity of its graphical model. It is also possible that the use of a measure of cross-entropy, such as the Kullback–Leibler divergence [15], might permit the learning of models which simultaneously minimize the intra-class entropy, while maximizing inter-class entropy. More research is needed on the subject of entropy thresholding in random graph clustering.

While the use of entropy as a clustering criterion has been shown to be effective, the precise relationship between the entropy of a distribution and its statistical properties is not well understood. It is possible that alternate distance measures, such as divergence or JM-distance [16], could be effectively applied to the clustering problem. Such distance metrics have precise statistical interpretations, and might allow for the establishment of theoretical bounds on the expected error of learned graphical models.

6. Summary

First order random graphs as introduced by Wong are a promising tool for structure–based classification. Their complexity, however, hampers their practical application. This complexity arises from the use of discrete distributions, and the subsequent need to estimate and update the entropy estimates of such distributions. We describe an extension to first order random graphs which uses continuous Gaussian distributions to model the densities of all random elements in a random graph. These First Order Gaussian Graphs (FOGGs) are shown to have several nice properties which allow for fast and efficient clustering and classification. Specifically, we show how the entropy of a FOGG may be computed directly from the Gaussian parameters of its random elements. This allows for fast and memoryless computation of the objective function used in the clustering procedure used for learning a graphical model of a class. Moreover, the use of continuous, parametric distributions eliminates the need to discretize continuous samples. We give a comparative evaluation between FOGGs and several traditional statistical classifiers. On our example problem, selected from the area of document analysis, our first order Gaussian graph classifier significantly outperforms statistical, feature-based classifiers. The FOGG classifier achieves a classification accuracy of approximately 98%, while the best statistical classifiers only manage approximately 91%. We also show that FOGGs are capable of representing structural sub-classes within the original classes. This feature allows FOGGs to capture structural variation that statistical classifiers are incapable of modeling.

References

[1] A.K.C. Wong, J. Constant, M.L. You, Random graphs, in: H. Bunke, A. Sanfeliu (Eds.), Syntactic and Structural Pattern Recognition: Theory and Applications, World Scientific, Singapore, 1990, pp. 179–195.

[2] H.-Y. Kim, J.H. Kim, Hierarchical random graph representation of handwritten characters and its application to hangul recognition, Pattern Recognition 34 (2001) 187–201.

[3] R. Alquézar, A. Sanfeliu, F. Serratosa, Synthesis of function-described graphs, in: Proceedings of the Joint IAPR International Workshops SSPR'98 and SPR'98, no. 1451 in Lecture Notes in Computer Science, Springer, Berlin, 1998, pp. 112–121.

[4] F. Cesarini, E. Francesconi, M. Gori, G. Soda, A two level knowledge approach for understanding documents of a multi-class domain, in: The Proceedings of the International Conference on Document Analysis and Recognition, 1999, pp. 135–138.

[5] C.K. Shin, D.S. Doermann, Classification of document page images based on visual similarity of layout structures, in: Proceedings of the SPIE Document Recognition and Retrieval VII, 2000, pp. 182–190.

[6] D. Doermann, A. Rosenfeld, E. Rivlen, The function of documents, in: Proceedings of the ICDAR97, 1997.

[7] T. Pavlidis, Structural Pattern Recognition, Springer, Berlin, 1977.

[8] L. Miclet, Structural Methods in Pattern Recognition, Springer, Berlin, 1986.

[9] H. Bunke, Recent developments in graph matching, in: ICPR00, Vol. II, 2000, pp. 117–124.

[10] A.D. Bagdanov, M. Worring, Fine-grained document genre classification using first order random graphs, in: Proceedings of the ICDAR2001, 2001, pp. 79–83.

[11] C.E. Shannon, A mathematical theory of communication, Bell System Techn. J. 27 (1948) 623–656.

[12] M. Gondran, M. Minoux, Graphs and Algorithms, Wiley, New York, 1984.

[13] M. de Berg, M. van Kreveld, M. Overmars, O. Schwarzkopf, Computational Geometry, Algorithms and Applications, Springer, Berlin, 1997.

[14] S.K. Murthy, S. Kasif, S. Salzberg, A system for induction of oblique decision trees, J. Artif. Intell. Res. 2 (1994) 1–32.

[15] S. Kullback, Information Theory and Statistics, Wiley, New York, 1959.

[16] P.H. Swain, R.C. King, Two effective feature selection criteria for multispectral remote sensing, in: Proceedings of the Third International Joint Conference on Pattern Recognition, 1973.

About the Author—ANDREW BAGDANOV received his bachelors and masters degrees in Mathematics and Computer Science from the University of Nevada, Las Vegas, where he was a member of the Information Science Research Institute. He is currently a Ph.D. student in computer science at the University of Amsterdam, working in the field of multimedia information analysis. His research interests include document understanding, pattern recognition, image processing, and functional programming languages.

About the Author—MARCEL WORRING received his masters degree (honors) and doctoral degree, both in computer science, from respectively the Free University Amsterdam ('88), and the University of Amsterdam ('93), The Netherlands. He currently is an assistant professor at the University of Amsterdam. His interests are in multimedia information analysis. Among other activities, he is project leader of MIA, a large project covering knowledge engineering, language processing, image and video analysis, and information space interaction, conducted in close relation with industry. In 1998 he was a visiting research fellow at the University of California, San Diego. He has published over 50 scientific publications and serves on the program committee of several conferences.