Graph Generative Models from Information Theory

Lin Han

A Thesis Submitted for the Degree of Doctor of Philosophy

the University of York

Departments of Computer Science

October 2012

Abstract

Generative models are commonly used in statistical pattern recognition to describe the probability distributions of patterns in a vector space. In recent years, sustained by the wide range of mathematical tools available in vector space, many algorithms for constructing generative models have been developed. Compared with the advanced development of the generative model for vectors, the development of a generative model for graphs has had less progress. In this thesis, we aim to solve the problem of constructing the generative model for graphs using information theory.

Given a set of sample graphs, the generative model for the graphs we aim to construct should be able to not only capture the structural variation of the sample graphs, but to also allow new graphs which share similar properties with the original graphs to be generated. In this thesis, we pose the problem of constructing a generative model for graphs as that of constructing a supergraph structure for the graphs.

In Chapter 3, we describe a method of constructing a supergraph-based generative model given a set of sample graphs. By adopting the *a posteriori* probability developed in a graph matching problem, we obtain a probabilistic framework which measures the likelihood of the sample graphs, given the structure of the supergraph and the correspondence information between the nodes of the sample graphs and those of the supergraph. The supergraph we aim to obtain is one which maximizes the likelihood of the sample graphs. The supergraph is represented here by its adjacency matrix, and we develop a variant of the EM algorithm to locate the adjacency matrix that maximizes the likelihood of the sample graphs. Experimental evaluations demonstrate that the constructed supergraph performs well on classifying graphs.

In Chapter 4, we aim to develop graph characterizations that can be used to measure the complexity of graphs. The first graph characterization developed is the von Neumann

entropy of a graph associated with its normalized Laplacian matrix. This graph characterization is defined by the eigenvalues of the normalized Laplacian matrix, therefore it is a member of the graph invariant characterization. By applying some transformations, we also develop a simplified form of the von Neumann entropy, which can be expressed in terms of the node degree statistics of the graphs. Experimental results reveal that effectiveness of the two graph characterizations.

Our third contribution is presented in Chapter 5, where we use the graph characterization developed in Chapter 4 to measure the supergraph complexity and we develop a novel framework for learning a supergraph using the minimum description length criterion. We combine the Jensen-Shanon kernel with our supergraph construction and this provides us with a way of measuring graph similarity. Moreover, we also develop a method of sampling new graphs from the supergraph. The supergraph we present in this chapter is a generative model which can fulfil the tasks of graph classification, graph clustering, and of generating new graphs. We experiment with both the COIL and "Toy" datasets to illustrate the utility of our generative model.

Finally, in Chapter 6, we propose a method of selecting prototype graphs of the most appropriate size from candidate prototypes. The method works by partitioning the sample graphs into two parts and approximating their hypothesis space using the partition functions. From the partition functions, the mutual information between the two sets is defined. The prototype which gives the highest mutual information is selected.

Contents

List of Figures

List of Tables

List of Symbols

$G_i = (V_i, E_i)$	Sample graph G_i with node-set V_i and edge-set E_i
$\|V_i\|$	Number of nodes in the graph G_i
$\Gamma = (V_\Gamma, E_\Gamma)$	Supergraph with node-set V_Γ and edge-set E_Γ
$\|V_\Gamma\|$	Number of nodes in the supergraph Γ
$\mathbf{A}$	Adjacency matrix of a sample graph
$\mathbf{M}$	Adjacency matrix of the supergraph
$\mathbf{D}$	Degree matrix
$\mathbf{L}$	Laplacian matrix
$\hat{\mathbf{L}}$	Normalized Laplacian matrix
$\hat{\lambda}$	Eigenvalue of the normalized Laplacian matrix
$\mathbf{S}$	Assignment matrix
$\mathcal{G}$	Sample-graph set
$\|\mathcal{G}\|$	Number of graphs in $\mathcal{G}$
$\mathcal{S}$	Assignment-matrix set
$H(\cdot)$	Von Neumann entropy
$\bar{H}(\cdot)$	Simplified von Neumann entropy
μ, K_a, B_a	Parameters in the probabilistic framework

Acknowledgements

Firstly, I would like to express my sincere gratitude to my responsible and resourceful supervisors, Professor Edwin R. Hancock and Professor Richard C. Wilson. I am very grateful to them for their valuable guidance and the time and energy they have put into every stage of my PhD study over the last four years. In addition, I highly appreciate the SIMBAD project funding offered to me by my supervisors, which has allowed me to undertake my PhD study.

Secondly, I would like to thank my internal examiner, Dr. Adrian Bors, for evaluating me research work and giving me lots of constructive suggestions at the different milestones in my PhD study.

There are some researchers who have provided valuable help with my research. In this respect, I would like to thank Luca Rossi and Dr. Andrea Torsello for providing the implementation code for the importance sampler. I would also like to thank them for their constructive suggestions on the work in this thesis.

My sincere thanks also go to all my friends in York, especially to my friends Weiping Xu and Lichi Zhang. I will remember the days spent together with the friends here for ever.

Finally, I would like to say thank you to my parents, it is their love that has supported and encouraged me to finish my PhD study. I would like to dedicate this thesis to my parents.

Declaration

I declare that all the work in this thesis is solely my own except where attributed and cited to another author. Most of the material in this thesis has been previously published by the author. Below is a complete list of publications on which Chapters 3 to 6 are based.

Journal Paper

- Lin Han, Francisco Escolano, Edwin R. Hancock and Richard C. Wilson. Graph Characterizations from von Neumann Entropy. *Pattern Recognition Letters*, 33(15): 1958–1967, 2012.

Conference Papers

- Lin Han, Richard C. Wilson, Edwin R. Hancock, Lu Bai and Peng Ren. Sampling Graphs from A Probabilistic Generative Model. To appear in *the International Conference on Pattern Recognition*, 2012.

- Lin Han, Luca Rossi, Andrea Torsello, Richard C. Wilson and Edwin R. Hancock. Information Theoretic Prototype Selection for Unattributed Graphs. To appear in *the Joint IAPR International Workshops on Structural and Syntactic Pattern Recognition and Statistical Techniques in Pattern Recognition*, 2012.

- Lu Bai, Edwin R. Hancock, Peng Ren and Lin Han. Graph Clustering Using Graph Entropy Complexity Traces. To appear in *the International Conference on Pattern Recognition*, 2012.

- Lin Han, Edwin R. Hancock and Richard C. Wilson. Entropy versus Heterogeneity for Graphs. *In proceedings of the IAPR-TC-15 International Workshop on Graph-Based Representations in Pattern Recognition*, pages 32–41, 2011.

- Lin Han, Edwin R. Hancock and Richard C. Wilson. Learning Generative Graph Prototypes Using Simplified von Neumann Entropy. *In proceedings of the IAPR-TC-15 International Workshop on Graph-Based Representations in Pattern Recognition*, pages 42-51, 2011.

- Na Li, Edwin R. Hancock, Xiao S. Zheng and Lin Han. Improved Content-based Watermarking Using Scale-invariant Feature Points. *In Proceedings of the International Conference on Image Analysis and Processing*, pages 636–649, 2011.

- Lin Han, Edwin R. Hancock and Richard C. Wilson. Characterizing Graphs Using Approximate von Neumann Entropy. *In proceedings of the Iberian Conference on Pattern Recognition and Image Analysis*, pages 484–491, 2011.

- Lin Han, Edwin R. Hancock and Richard C. Wilson. An Information Theoretic Approach to Learning Generative Graph Prototypes. *In Proceedings of the International Workshop on Similarity-Based Pattern Analysis and Recognition*, pages 133–148, 2011.

- Lin Han, Richard C. Wilson and Edwin R. Hancock. A Supergraph-based Generative Model. *In Proceedings of the International Conference on Pattern Recognition*, pages 1566–1569, 2010.

Chapter 1

Introduction

In this chapter we provide an introduction for the research work presented in the thesis. We commence by introducing the problems encountered in learning graph data, followed by a description of our research goals and, finally, we provide an outline of the thesis at the end of this chapter.

1.1 The Problems

Relational graphs provide a convenient means of representing structural patterns. Examples include the arrangement of shape primitives or feature points in images, molecules and social networks. When abstracted in this way, complex data can be compared or matched using graph matching techniques. Although matching problems such as subgraph isomorphism or inexact graph matching are computationally expensive, there are a number of effective algorithms based on probabilistic [25], optimization [48] or graph-spectral [93] techniques that can give reliable results in polynomial time.

However, despite considerable progress in the problems of representing and matching data using graph structures, dealing with graph data is still a long-standing problem. There are two reasons why graphs are more difficult to manipulate than pattern vectors. One is

that there is no canonical ordering for the nodes in a graph, so correspondence between nodes must be established as a prerequisite [107]. The other is that the variation in graphs of a particular class may manifest itself as subtle changes in the structure, i.e. variations in a) node or edge attributes, b) node or edge composition and c) edge connectivity. For instance, the number of nodes and edges in a graph may be different from other graphs in the same class. Thus, even if the nodes or the edges of graphs could be encoded in a vectorial manner, the vectors would be of variable length [107].

The reasons above render the difficulty in the analysis of graph data for the purpose of characterizing graphs. Unlike pattern vectors, when the analysis of graph data is attempted, there is frequently no labeling or ordering of the nodes of the structure to hand. For the graph characterizations which require reliable node correspondences, they can prove very time consuming and even fragile, since they invariably require inexact graph matching over the dataset. It is for this reason that the use of permutation invariant graph characterizations has proved to be an attractive one. Although there are a number of simple alternatives that can be used, such as node or edge frequency, edge density, diameter and perimeter, these have proved to be ineffective as a means of characterizing variations. Instead, it has proved necessary to resort to more complex representations. One of the most successful of these has been to use graph-spectral methods [64][107]. Here the distribution of the eigenvalues and eigenvectors can be used to construct permutation invariants that do not require node correspondences. Unfortunately, the graph spectral method can prove to be computationally burdensome. The reason for this is that the computation of the graph-spectrum is cubic with regard to the number of nodes.

Another resultant difficulty is the construction of a generative model for graphs that captures structural variations present in the sample set. Compared with the advanced development of graph matching algorithms, the issue of how to capture variability in such representations has received relatively little attention. By contrast, there is a wealth of literature on how to construct statistical generative models that can deal with quite complex

data for vectorial patterns, including those arising from the analysis of variability in shape [29][76][57]. The lack of progress in graph generative models is due to the difficulty in developing representations that capture variations in graph structure. As previously mentioned, there are three types of graph structural variation. Of the three, the problem of learning edge connectivity is probably the most challenging. Broadly speaking, there are two approaches to characterizing variations in edge structure for graphs. The first of these is graph spectral, while the second is probabilistic. In the case of graph spectra, many of the ideas developed in the generative modeling of shape using principal components analysis can be translated relatively directly to graphs using simple vectorization procedures based on the correspondences conveyed by the ordering of Laplacian eigenvectors [64]. Although these methods are simple and effective, they are limited by the lack of stability of the Laplacian spectrum under perturbations in graph structure. The probabilistic approach is potentially more robust, but requires accurate correspondence information to be inferred from the available graph structure.

1.2 Our Goals

The goals of this thesis are to explore efficient graph characterizations and, with the help of the derived characterizations, construct a generative model for graphs. In this thesis we focus on the problem of capturing edge connectivity variations and aim to develop a generative model that can be used to describe structural variations of edge connectivity in the sample graphs. Specifically,

a) We aim to explore more efficient graph characterizations. To this end, we turn to information theory and use entropy to define measures of graph characterizations. In particular, we will investigate the von Neumann entropy of graphs, which relates to the eigenvalues of the normalized Laplacian matrix as a graph characterization. Using the von Neumann entropy, we will explore whether we can approximate the entropy in terms

of node degree statistics and obtain a simplified form, the computational complexity of which is much lower.

b) We aim to take an information theoretic approach to construct a generative model for graphs. Once we have the entropy based graph characterizations developed in the last step, we will use them to measure the complexity of the generative model and construct a generative model that trades off goodness-of-fit by adopting the minimum description length criterion. Moreover, we seek a generative model which is multi-functional and which can be used to classify graphs, measure graph similarity and also to generate new sample graphs.

1.3 Thesis Outline

Having described the overall goals of the thesis, we proceed to outline the structure of the thesis. In Chapter 2, we reviews the literature relevant to the research described in this thesis.

In Chapter 3, we present a novel method of constructing a supergraph-based generative model for a set of graphs. We pose the problem of constructing a generative model for graphs as that of learning a supergraph structure which can capture the edge connectivity variations present in the sample graphs. We experiment with a real world dataset and investigate its performance in classifying graphs.

In Chapter 4, we illustrate how the von Neumann entropy can be used as a measure of graph characterization and, moreover, we also develop its simplified form. In the experimental part, we evaluate these two graph characterizations and compare them with alternative graph characterizations .

In Chapter 5, we combine the methods previously developed in Chapter 3 and Chapter 4 to construct a generative prototype for graphs by adopting a minimum description length approach. A variant of the expectation-maximization algorithm is developed to minimize

the overall description length criterion. We also develop new mechanisms so that the generative model is capable of measuring graph similarity and of generating new samples. Experimental investigations reveal the utility of our generative model.

In Chapter 6, a prototype graph size selection method is provided. We extend the theory of approximate set coding from the vector domain to the graph domain and show how the problem of prototype size selection can be solved by optimizing the mutual information between two partitioned sets of sample graphs.

In the final chapter, we offer some conclusions, including a summary of the contributions we have made and directions for future research.

Chapter 2

Literature Review

In this chapter, we will review the literature relevant to our work described in the thesis. The two main aims of the thesis are to explore efficient methods to characterize graphs and to use the derived graph characterization to construct a generative model for graphs that can capture graph structural variations. To comply with these aims, we partition the content of the chapter into six parts. We commence in Section 2.1 by introducing the graph representation. We then review the spectral graph theory and its applications in the area of image segmentation and graph matching in Section 2.2. We survey graph characterizations in Section 2.3. We review generative models for graphs in Section 2.4, followed by a review of deep learning in Section 2.5. Finally, in Section 2.6, we review some measures from information theory that we will use to develop our methods in the following chapters.

2.1 Graph Representation

The graph-based representations have been widely used with considerable success in the problems of shape representation [3], segmentation [40], matching [73], and object recognition [112] in computer vision since relational graphs as abstractions for pictorial infor-

mation were first demonstrated by Barrow and Burst [8], and Fishchler and Elschlager [43]. In the cases of the genomics and networks, we can naturally represent the data as structural graphs. However, it is not that straightforward when encountered with image or scene data. Dealing with these data using graph-based methods requires converting them to graph representation and this involves extracting feature points on images and arranging the feature points to graphs. In the graph representation of these data, the extracted features are represented as graph nodes and their arrangement are represented by an edge structure.

To represent the images in graphs, we need to arrange the set of extracted feature points in a way that can preserve their general layout. An issue to be noted is that we need to have a distance measure between feature points before we construct graph representation for the feature points. The distance between feature points can be defined in many ways. It can be defined as the Euclidean distance between the descriptors of the feature points or the Euclidean distance between the locations of the feature points or one combining both. After we have the pairwise distance of the feature points, we proceed to the graph construction step. There are many different methods to connect these feature points in graphs. A famous one among them is the Delaunay triangulation invented by Boris Delaunay [31] in 1934. The Delaunay triangulation of the feature points has such representation that no feature point is inside the circumcircle of any triangle of other points. A property of the Delaunay triangulation is that it maximizes the minimum angle of all the angles of the triangles in the triangulation [94]. Other graph representations include the Gabriel graph [47] and the K-nearest neighbour graphs [72]. In the Gabriel graph, two points are connected by an edge when there are no other points in the circle whose diameter is the line segment jointing the two points. The nearest neighbor graph representation, as indicated by its name, connects each node to its K-nearest neighbour nodes.

2.2 Spectral Graph Theory

Spectral graph theory [13] [84] [26] is a branch of mathematics which studies the structural properties of a graph by exploring the eigensystem of the graph. The eigensystem of a graph consists of the eigenvalues and eigenvectors of an associated matrix of the graph, such as its adjacency matrix or Laplacian matrix (the degree matrix minus the adjacency matrix). The eigenvalues, ordered in terms of their magnitude, constitute the spectrum of the graph. An important property of the spectrum is that it is invariant to the labelling of the graph when the graph is non-attributed. The subject of spectral graph theory has acquired considerable topicality because spectral graph theory is very useful for solving problems of image segmentation and graph matching.

Alternative methods based on the eigensystem have been used to solve the problems of pairwise clustering and image segmentation. Some of the earliest work was done by Scott and Longuet-Higgins in [88]. They build an proximity matrix to measure the dissimilarities between image features and then use the eigenvalues and eigenvectors of the proximity matrix to partition features into clusters. Thereafter, Shi and Malik [92] treated image segmentation as a graph partitioning problem and introduced the normalized cut criterion to segment graphs. To optimize this criterion, they develop a generalized eigenvalue system in which they iterated using the eigenvector with the second smallest eigenvalue of the affinity matrix to bipartition the graph. Examples also include those described in [96] [103].

With regard to the problem of the graph matching, there are lots of examples of the application of spectral matching methods. In the pioneer work of Umeyama [101], he employed an analytic approach to the optimum matching problem of weighted graphs and efficiently found a permutation matrix close to the optimum one by taking the outer product of the left eigenvector matrices for the two graphs. In related work, Shapiro and Brady [91] have proposed a method for recovering point-feature correspondence by using the eigenvectors of a proximity matrix that records the Gaussian weighted distance

between features within the shapes. However, both methods are exact graph matching algorithms and they can only deal with graphs of the same size (the same number of nodes).

Luo and Hancock [61] have described an efficient algorithm for inexact graph matching that can accommodate graphs of different sizes. In their work, they develop a probabilistic framework to measure graph similarity and pose the problem of graph matching as maximum likelihood estimation using the apparatus of the EM algorithm. In the recovery of the correspondence matching, they ingeniously cast the problem in a matrix framework which can be efficiently solved using singular value decomposition.

In addition, spectral graph theory also provides approaches to measuring graph distance. For instance, Wilson and Zhu [108] have used the Euclidean distance between spectra of graphs to measure the distance of graphs in classification and clustering tasks.

Many concepts in spectral graph theory, such as the heat kernel, commute time and random walks, play important roles in analyzing graphs. Heat kernels of graphs are widely used as a means of characterizing graphs, clustering graphs and embedding graphs [111] [6] [5]. Besides the utility for graph clustering and embedding [85][11], the commute time and random walks also have applications for image segmentation and multi-body motion tracking [74][50].

2.3 Graph Characterizations

Graph characterizations are of vital importance in the analysis of graph data. Broadly speaking, these characterizations falls into two groups. The first are permutation invariant characteristics extracted from the graph structure and the others require having the node correspondence to hand [41][42]. The second type of graph characterizations usually involves applying graph matching algorithms to obtain the node correspondence and thus their performance relies on the goodness of these matching algorithms. Therefore, the use

of permutation invariant graph characteristics has proved to be more attractive.

Examples of the invariant graphs characterizations include Laplacian spectra and characteristic polynomials of elements of the spectral matrix [62] [107]. Luo *et al.* [62] have used the ordered eigenvalues from the Laplacian matrices of graphs as graph features to perform graph clusterings. Wilson *et al.* [107] have used the elements of the Laplacian matrices of graphs to construct symmetric polynomials that are permutation invariants. The coefficients of these polynomials can be encoded in a vector manner and used as graph features. Xiao *et al.* [111] have taken the study of spectral graph invariants one step further. In their studies, they perform an analysis of the heat kernel for graphs, and show that the Riemann zeta function can be used to generate a number of powerful invariants from the normalized Laplacian spectrum.

Recently, graph characterizations that can quantify the intrinsic complexity of graphs and networks have attracted significant attention due to their fundamental practical importance, not only in network analysis [38] but also in other areas such as pattern recognition and control theory. Some of the existing quantifications are easily computable, i.e. they have polynomial computational complexity [37] [9], but others are not since they rely on NP-hard problems and are computationally intractable. These existing approaches are based on notions of either randomness complexity or statistical complexity.

Randomness complexity aims to quantify the degree of randomness or disorganization of a combinatorial structure. This approach aims to characterize an observed graph structure probabilistically and to compute its associated Shannon entropy. Escolano *et al.* [34] have constructed a graph complexity measure using the entropies associated to the Birkhoff-von Neumann decomposition on the heat kernel of the graph. In their subsequent studies, they extended their work by defining the heat flow complexity measure and the corresponding heat flow based thermodynamic depth measure [36].

Statistical complexity, on the other hand, aims to characterize a combinatorial structure using statistical features such as node degree statistics, edge density or the Laplacian

spectrum. Early examples of the network irregularity indices falling into this category include the index proposed by Collatz and Sinogowitz [27], which is defined as the difference between the principal (largest) eigenvalue of the adjacency matrix and the average node degree, and the Bell's index, which is the variance of nodes degree [9]. Recently, Estrada [37] has defined an index that accounts for the heterogeneity of networks. To compose this index, he starts by defining a local index which is a function of the node degree to measure the irregularity of a single link (edge) in the network. The heterogeneity index of a network proposed is obtained as the sum of the link irregularity for all links in the network. By choosing a suitable function, this index can be expressed as a quadratic form of the Laplacian matrix of the network. Passerini and Severini [70] have shown how to use the von Neumann entropy to measure network irregularity.

Viewed historically, most early work in this area falls into the randomness class, while recent work is statistically based. The main drawback of randomness complexity is that it does not properly capture the correlations between vertices [39]. Statistical complexity aims to overcome this problem by measuring regularities beyond randomness, and does not necessarily grow monotonically with randomness.

2.4 Generative Models of Graphs

In this section, we discuss the work of constructing generative models for graphs. There are three types of graph structural variations, namely variations in a) node or edge attributes, b) node-composition and c) edge-connectivity, which provide a natural framework for analyzing the state-of-the-art in the literature. Most of the literature can be viewed as modeling variations in node or edge attributes. In fact, most of the work on Bayes nets in the graphical models literature falls into this category [45] [22] [46]. The Bayes nets used are a graph-based representation of a multivariate joint probability distribution that exploits the dependencies or independencies between variables. These Bayes

nets can be used to do diagnosis, learning, explanation, and many other inference tasks necessary. Thus they have wide applications in the area of genetics, social science and computer science. There are also some well documented studies in the structural pattern recognition literature that also fall into this category, including the work of Wong *et al.* [109], Bagdanov and Worring [4]. Wong *et al.* [109] have introduced a first order random graphs for structural-based classification. In their random graph model, the vertices and edges are associated with discrete random variables taking values over the attribute domain of the graphs. However, the use of the discrete densities complicates the learning and classification process and hampers the practical application. Later, Bagdanov and Worring [4] extended the first order random graphs by using continuous Gaussian distributions to model the densities of random variables in the graphs. Their method overcomes some of the computational difficulties and allows for fast and efficient clustering and classification.

The problems of modeling variations in node and edge composition are more challenging, since they focus on modeling the structure of the graph rather than its attributes. For the restricted class of trees, Torsello and Hancock [99] have built a tree union to cluster trees. In their clustering method, the correspondences between nodes are unknown and must be inferred as part of the learning process. They use a minimum description length approach to fitting the tree union to graph data. The node composition is recovered by minimizing the edit distance which is linked to the description length criterion. Since trees are rooted, the learning procedure is facilitated and can be performed in polynomial time. However, this greedy strategy does not translate tractably to graphs where the complexity becomes exponential. Torsello and Dowe [98] have recently made some progress in extending this method to graphs using importance sampling techniques [97] to overcome some of the computational bottlenecks.

The problem of learning edge-connectivity is probably the most challenging of those listed above. The literature on characterizing variations in edge structure for graphs can be

categorized into two types. The first of these are graph spectral approaches, while the second are probabilistic approaches. The graph spectral approaches are developed by using the eigenvalues and eigenvectors of the associated graph matrices from graph spectral theory. Xiao and Hancock [110] have explored how to use the eigenvalues and eigenvectors from the heat kernel matrix to construct a generative model for graphs. They first embed the nodes of graphs into a vector space by performing the Young-Householder decomposition on the heat kernel matrix, and then describe the distribution of the coordinates of the nodes using a Gaussian distribution. Although the variations in graph structure can be adequately captured by the covariance matrix of the embedded node coordinates, it is difficult to reconstruct graphs from these representations. White and Wilson [104] have proposed a different spectral generative model. They create separate distributions for eigenvalues and eigenvectors, from which they can generative a new matrix that is close to a Laplacian matrix of a graph. Through setting a threshold, the Laplacian matrix can be recovered back to an adjacency matrix, which gives the structure of the graph. Therefore, their method is an improvement in the sense that their model can generate new graph structures. Although the methods based on the spectral graph theory are simple and effective, they are limited by the stability of the eigensystems of the graphs under perturbations in graph-structure.

The probabilistic approaches, on the other hand, are potentially more robust. An example of the approach has been developed by Luo *et al.* [63], where the authors directly convert graphs into long vectors by stacking the elements of the adjacency matrices of graphs, and exploit the structural variations of graphs by constructing a linear deformable model. Before stacking the elements of the adjacency matrices, however, they need to align the graphs so that the nodes are in the same order. They use the algorithm in [61] to obtain the node correspondence information. The drawback of probabilistic approaches is that they require accurate correspondence information to be inferred from the available graph structure before constructing the statistical models. To date, the most effective

algorithm falling into this category exploits a part-based representation [105]. In the part-based representation, graphs are represented by a clustering of subgraphs. The variations in graphs are modelled by observing which subgraphs are present in each graph and how these subgraphs are connected. Because the model defines a distribution based on the presence of subgraphs and the way subgraphs are connected, new graphs can be sampled from the distributions.

2.5 Deep Learning

Recently, a new area of machine learning called deep learning emerged and has attracted considerable interest. The research in this area advocates learning multiple levels of representation in order to model complex relationships among data. High level features and concepts are defined in terms of lower-level ones, and this hierarchical representation is called deep architecture. Before 2006, attempts at training deep architectures (mostly neural networks) failed, with the exception of shallow neural networks with one or two hidden layers. In 2006, Hinton's revolutionary work on deep belief networks [55] made a breakthrough in learning deep architectures. The main breakthrough made by Hinton *et al.* is that they develop a greedy, layer-by-layer unsupervised learning algorithm that allows efficient training of the deep belief networks [86]. With the help of the algorithm, the deep belief networks form probabilistic generative models, which consist of multiple layers of variables. The top layer consists of the observed variables and the remaining layers consist of hidden variables. The variables in a lower layer control the variables in the upper layers. The main building block of a deep belief network is a Restricted Boltzmann Machine (RBM). The RBM is a stochastic neural network, which consists of one layer of visible variables (neurons) and one layer of hidden variables (neurons). Variables in each layer have no connections between them and are connected to all variables in the other layer. Connections between variables are undirected, which means that information

flows in both directions during the training and during the usage of the network.

Since then, the deep generative model has been applied with success in many tasks. In the work of [55], a deep belief network is used to learn a generative model of the joint distribution of handwritten digit images and their labels. This generative model gives better digit classification than the best discriminative learning algorithms. Examples also include the work reported in the area of natural language processing [28], where a deep neural networks can facilitate multitask learning (i.e. given a sentence, outputting a host of language processing predictions such as part-of-speech tags, chunks and named entity tags) and semi-supervised learning, both of which are able to improve the generalization of the shared tasks and result in state-of-the-art performance.

Ranzato *et al.* [76] have used a deep belief network to improve a gated Markov Random Field (MRF) generative model on images. The gated MRF generative model is composed of two hidden layers, one set of hidden variables is used to create an image-specific model of the covariance structure of the pixels and the other set of hidden variables is used to model the intensities of the pixels. Their deep belief network uses the gated MFR as the lowest level and adds several layers of Bernoulli hidden variables to model the statistical structure in the hidden activities of the gated MRF. Their experiments have shown that the deep belief network is better than the gated MRF model at generating high-resolution natural images, and that the features that it learns are good at discriminating facial expressions or scene images.

In most of the methods that adopt the deep learning to train probabilistic distributions of the observed data, such as images and sentences, variables in the hierarchical structure have vector value. Therefore those methods closely relate to the generative models of graphs that model the distributions of node and edge attributes.

2.6 Information Theory Related to Our Work

Most of our work presented in the thesis relates to information theory. The related information theory includes the von Neumann entropy, the minimum description length criterion and the mutual information. We review the three concepts in this section.

2.6.1 Von Neumann Entropy

The von Neumann entropy was introduced by John von Neumann to measure irreversibility processes in quantum statistical mechanics [102]. It is an extension of the Gibbs entropy and the Shannon entropy to the quantum realm. The von Neumann entropy is defined as entropy of the density matrix of a quantum system. In quantum mechanics, a quantum system is described by state vector $|\psi\rangle$. If a quantum system has only one single state vector, it is then called pure state. In most general cases, the quantum systems have a mixed quantum state. A mixed quantum state corresponds to a set of state vectors $|\psi_j\rangle$ with different probabilities η_j. The probabilities η_j satisfy the condition that $0 \leq \eta_j \leq 1$ and $\sum_j \eta_j = 1$. The density matrix of the quantum system is

$$\rho = \sum_j \eta_j |\psi_j\rangle\langle\psi_j|, \tag{2.1}$$

where $|\psi_j\rangle$ is a column vector and $\langle\psi_j|$ is the transpose of $|\psi_j\rangle$. The density matrix ρ defined above has the following properties. Its eigenvalues are non-negative and its trace sums up to one $\text{Tr}(\rho) = 1$. Given the density matrix, the von Neumann entropy is [10][102]

$$H(\rho) = -\text{Tr}(\rho \ln \rho). \tag{2.2}$$

To compute $\ln \rho$, we perform $\rho = \mathbf{\Phi}(\ln \mathbf{\Lambda})\mathbf{\Phi}^T$. $\mathbf{\Phi}$ is a matrix whose columns are eigenvectors of ρ and $\ln \mathbf{\Lambda}$ is a diagonal matrix whose diagonal line has elements which are logarithms of the eigenvalues of ρ. The von Neumann entropy is equal to

$$H(\rho) = -\sum_j \lambda_j \ln \lambda_j \,, \tag{2.3}$$

where λ_j is the eigenvalue of the density matrix.

Since the entropy is defined for a quantum state, a mapping from graphs into states is required if we want to explore the von Neumann entropy associated with graphs. In the literature area, many methods have been proposed to map graphs into quantum states. Examples include the work in [19][26]. Recently, this research has been taken further by Passerini and Severini [70], who build a faithful mapping between the Laplacians and quantum states. They show that the density matrix of a graph can be obtained by scaling the (normalized) Laplacian matrix of the graph and from which the von Neumann entropy of graphs can be defined. In Chapter 4, we are going to explore the graph characterizations from the von Neumann entropy of graphs.

2.6.2 Minimum Description Length Criterion

Model selection is one of the most important problems in statistical inference. It deals with the problem of selecting the best underlying statistical models from a set of candidate models. The minimum description length criterion (MDL), introduced by Rissanen [79], is proposed to provide a solution to this problem. The minimum description length is a formalization of Occam's Razor and its basic idea is to select the model that can compress data most [51]. The earliest implementation of this idea is the two-part code version of the minimum description length criterion, which respectively encodes the data and model complexity and selects the best model by minimizing the sum of their code-length. The rationale of the two part version is that the complexity of the model is against goodness of the fit, which will automatically avoid overfitting and will have a good predictive performance on new data. However, a problem of this two-part version is that it is difficult to find a good code for the model. Later, Rissanen [81] sidestepped this problem by using a one-part version, which comes out to the refined MDL.

Torsello and Hancock [99] have adopted a two-part MDL to the problem of fitting a tree-union model, where they encode the complexity of tree-union in terms of the param-

eters in their model. Davies *et al.* [113] have described a method for building statistical shape models from a set of boundaries using the minimum description length approach. In their method, they pose the problem of building the shape model as one of finding the parameterizations for the correspondence points on the shapes. The parameterizations are selected as those which minimize the description length of the training set. Examples also include using minimum description length to evaluate the quality of business process models [24] and using the minimum description length principle to segment multilingual documents and identify the language of the segments [114].

2.6.3 Mutual Information

Since Shannon [90] introduced mutual information to measure the dependence between variables, there have been substantial theoretical and practical developments of the concept. For instance, Tourass *et al.* [100] have used the mutual information criterion to select the optimal subset of features in computer-aided diagnosis, where the mutual information between random variables (features) is estimated using the histogram approach. Examples also include using the maximum mutual information to train hidden Markov models [49] and applying the mutual information in medical image processing and image registration task [71]. Recently, Buhmann *et al.* [20][21] have proposed an information theoretic model selection theory called the approximate set coding where they develop a communication scenario to measure the generalization capacity of models. The generalization capacity of the models is defined using the mutual information between the coarsened training data and the coarsened test data. However, their model selection method is proposed in clustering in the vector domain. In Chapter 6, we will extend his theory to graph domains.

2.7 Conclusions

Based on the review of the related literature, we may draw several conclusions. First, although there is a substantial body of research on graph characterizations, developing efficient graph characterizations that can quantify the intrinsic complexity of graphs and networks is still an urgent problem. These existing graph characterization measures either suffer from the curse of expensive computational complexity or are not effective. In the thesis, we will explore the feasibility of extracting useful and efficient graph characterizations from the von Neumann entropy as graph complexity measures.

The second point derived from the literature review is that the method of learning generative models for graphs using information theory, under the guide of the minimum description length criterion, has not been proposed. It is of value to explore this area, since the generative models developed in this way can avoid the problem of overfitting and generalize well to new data. Developing such methods could be achieved with the help of a well-developed graph characterization measure, which can efficiently capture graph complexity. Later, we will show how we use the graph characterizations extracted from the von Neumann entropy of graphs to measure the complexity of the generative model and take an information theoretic approach to construct a generative model using the minimum description length criterion.

Thirdly, the review of the mutual information also suggests a method for selecting graph models. The recently developed theory of approximate set coding proposed a method of selecting models by maximizing the mutual information between two partitioned datasets. Although this theory is proposed for clustering in the vector domain, it provides scope for us to apply it to graphs. We will extend the theory to the graph domain and use it for selecting the sizes of the prototype graphs.

Chapter 3

A Supergraph-based Generative Model

3.1 Introduction

This chapter proposes a method of constructing generative model for a set of sample graphs. We follow Torsello and Hancock [99] and pose the problem of constructing the generative model as that of learning a supergraph structure which can describe the edge structural variations present in the set. The supergraph is a graph-union that can capture the structural variations of the graphs in the sample set. To furnish the required learning framework, we use the probabilistic framework developed by Luo and Hancock [61] to describe the distribution of the sample graphs. The structure of supergraph we aim to learn is the one that maximizes the likelihood of the sample graphs. To locate the structure of this supergraph, we develop a variant of the expectation-maximization (EM) algorithm where both the structure of the supergraph and the correspondences between the nodes of the sample graphs and those of the supergraph are treated as missing data. This novel technique is applied to a database of object views, and used to learn class prototypes that can be used for the purposes of recognition.

The main contribution of this chapter is that by extending the work of Luo and Hancock [61], we develop a novel generative model for a set of graphs based on a supergraph

structure. The supergraph here is the one that maximizes the likelihood of the sample graphs. The second contribution is that we develop a variant of EM algorithm to realize the maximum-likelihood estimation. The outline of the chapter is as follows. In Section 3.2 we review the likelihood function developed by Luo and Hancock [61]. This likelihood function will later be used to formulate our probabilistic framework. In Section 3.3, we describe the methodology we use to learn the supergraph structure. The variant of the EM algorithm is also provided here. In Section 3.4 we give some experimental analyses. Finally, in Section 3.5 we draw our conclusions.

3.2 The Likelihood Function

Given a set of sample graphs, our aim is to learn a generative model that can be used to describe the distribution of the sample graphs and characterize the structural variations present in the set. Here we pose the problem as that of learning a supergraph. To commence our development we require a probabilistic framework to measure the likelihood of the sample graphs. We use the *a posteriori* probability developed by Luo and Hancock [61] to describe the likelihood function of the sample graphs. This *a posteriori* probability was initially developed to measure the similarity between a data graph and a model graph in a graph matching problem. In our problem we use it to measure the likelihood of a sample graph being generated from a supergraph. In this section we review how they construct the *a posteriori* probability. To make the content in this section consistent with the following sections, we explain the development of the *a posteriori* probability in the context of a sample graph and the supergraph.

To commence, we introduce some notations. We represent the sample graph by $G = (V, E)$ where $V = \{a, b, \dots\}$ represents the node-set in the graph and E represents the edge-set. The supergraph is denoted by $\Gamma = (V_\Gamma, E_\Gamma)$ with node-set $V_\Gamma = \{\alpha, \beta, \dots\}$ and edge-set E_Γ. The structure (edge connectivity) of the two graphs are indicated by their

adjacency matrices. The adjacency matrices are square matrices and the dimension of an adjacency matrix of a graph is equal to the number of the nodes in the graph. If two nodes in a graph are connected by an edge, the corresponding element in the adjacency matrix is one, otherwise it will equal zero. We denote the adjacency matrix of the sample graph G by $\mathbf{A}$ and the elements of the adjacency matrix are

$$A_{ab} = \begin{cases} 1 \text{ if } (a,b) \in E \\ 0 \text{ otherwise.} \end{cases} \tag{3.1}$$

Similarly, we represent the adjacency matrix of the supergraph Γ by $\mathbf{M}$ and have its elements

$$M_{\alpha\beta} = \begin{cases} 1 \text{ if } (\alpha,\beta) \in E_\Gamma \\ 0 \text{ otherwise.} \end{cases} \tag{3.2}$$

In the graph matching problem, the node correspondence information between the two graphs is represented by an assignment matrix $\mathbf{S}$ whose dimension is $|V| \times |V_\Gamma|$ where $|V|$ and $|V_\Gamma|$ are respectively the number of the nodes in the sample graph and those in the supergraph. The assignment matrix indicates the node correspondences between the sample graph G and the supergraph Γ. It has elements

$$s_{a\alpha} = \begin{cases} 1 & \text{if } f(a) = \alpha \\ 0 & \text{otherwise,} \end{cases} \tag{3.3}$$

where $f(a) = \alpha$ means that node $a \in V$ is matched to node $\alpha \in V_\Gamma$.

According to Luo and Hancock [61], the idea underpinning their developed likelihood function is that the node correspondences between the two graphs are hidden variables and the nodes in the sample graphs arise through a noisy observation process. That is to say, there is a possibility that any single node of the sample graph may be matched to any node in the supergraph. Therefore, to entertain this feature, the authors define the probability of observing a node in the sample graph in the form of a summation over the set

of all the possible correspondences. By assuming the nodes in the sample graph are independent, the likelihood function of the sample graph involves factorizing the observation probability over all the nodes in the sample graph, and is written

$$P(G|\Gamma, \mathbf{S}) = \prod_{a \in V} \sum_{\alpha \in V_\Gamma} P(a|\alpha, \mathbf{S}), \tag{3.4}$$

where $P(a|\alpha, \mathbf{S})$ is the probability that node a in the sample graph is in correspondence with node α in the supergraph under the assignment matrix $\mathbf{S}$.

They proceed to develop a model for the observation probability $P(a|\alpha, \mathbf{S})$. Using Bayes' theorem, $P(a|\alpha, \mathbf{S})$ is equal to

$$P(a|\alpha, \mathbf{S}) = \frac{P(\mathbf{S}|a, \alpha)P(a, \alpha)}{P(\mathbf{S}|\alpha)P(\alpha)} \, . \tag{3.5}$$

Assuming that the observation probability of the assignment matrix is factorizable over the set of the assignment variables, the above function becomes

$$P(a|\alpha, \mathbf{S}) = \frac{\{\prod_{b \in V} \prod_{\beta \in V_\Gamma} P(s_{b\beta}|a, \alpha)\}P(a, \alpha)}{\{\prod_{b \in V} \prod_{\beta \in V_\Gamma} P(s_{b\beta}|\alpha)\}P(\alpha)} \, . \tag{3.6}$$

Applying Bayes' theorem, they have

$$P(s_{b\beta}|a, \alpha) = \frac{P(a|\alpha, s_{b\beta})P(\alpha|s_{b\beta})P(s_{b\beta})}{P(a, \alpha)} \tag{3.7}$$

and

$$P(s_{b\beta}|\alpha) = \frac{P(\alpha|s_{b\beta})P(s_{b\beta})}{P(\alpha)} \, , \tag{3.8}$$

then the function can be rewritten as

$$P(a|\alpha, \mathbf{S}) = \frac{\{\prod_{b \in V} \prod_{\beta \in V_\Gamma} \frac{P(a|\alpha, s_{b\beta})P(\alpha|s_{b\beta})P(s_{b\beta})}{P(a,\alpha)}\}P(a, \alpha)}{\{\prod_{b \in V} \prod_{\beta \in V_\Gamma} \frac{P(\alpha|s_{b\beta})P(s_{b\beta})}{P(\alpha)}\}P(\alpha)} \, . \tag{3.9}$$

Canceling terms $P(\alpha|s_{b\beta})$ and $P(s_{b\beta})$ which appear both in the numerator and denominator and collecting together terms, they find the expression simplifies to

$$P(a|\alpha,\mathbf{S}) = [\frac{1}{P(a|\alpha)}]^{|V|\times|V_\Gamma|-1}\prod_{b\in V}\prod_{\beta\in V_\Gamma}P(a|\alpha,s_{b\beta}). \tag{3.10}$$

They further assume that nodes in the sample graph are conditionally dependent on the supergraph graph nodes only in the presence of the assignment matrix $\mathbf{S}$, then $P(a|\alpha) = P(a)$. Hence,

$$P(a|\alpha,\mathbf{S}) = B_a\prod_{b\in V}\prod_{\beta\in V_\Gamma}P(a|\alpha,s_{b\beta}), \tag{3.11}$$

where

$$B_a = [\frac{1}{P(a)}]^{|V|\times|V_\Gamma|-1}. \tag{3.12}$$

is a constant and its value depends only on the identity of the sample graph node a.

To develop a model for the conditional probability $P(a|\alpha,s_{b\beta})$, the authors draw on the work of Wilson and Hancock [106]. The idea behind the model is that a node α in the supergraph can emit a symbol a drawn from the nodes in the sample graph and the probability that this correspondence is correct is $1-P_e$, while the probability that it is in error is P_e. The correctness of the correspondence is gauged by checking whether nodes a and b in the sample graph are matched to a valid edge in the supergraph. $A_{ab}M_{\alpha\beta}s_{b\beta}$ is used for the test of edge-consistency. It has a unity value only when node b in the sample graph is matched to node β in the supergraph and they also satisfy $(a,b)\in E$ and $(\alpha,\beta)\in E_\Gamma$. When the condition is not met, the quantity is zero. Using this switching property and assuming the nodes in the sample graph are derived from the supergraph under a Bernoulli distribution, the condition probability is

$$P(a|\alpha,s_{b\beta}) = (1-P_e)^{A_{ab}M_{\alpha\beta}s_{b\beta}}P_e^{1-A_{ab}M_{\alpha\beta}s_{b\beta}}. \tag{3.13}$$

Substituting Equation (3.13) into Equation (3.11), they have

$$P(a|\alpha,\mathbf{S}) = B_a \prod_{b\in V}\prod_{\beta\in V_\Gamma}(1-P_e)^{A_{ab}M_{\alpha\beta}s_{b\beta}}P_e^{1-A_{ab}M_{\alpha\beta}s_{b\beta}}. \tag{3.14}$$

The above function can be expressed as a natural exponential function

$$P(a|\alpha,\mathbf{S}) = K_a \exp[\mu \sum_{b\in V}\sum_{\beta\in V_\Gamma} A_{ab}M_{\alpha\beta}s_{b\beta}], \tag{3.15}$$

where

$$\mu = \ln\frac{1-P_e}{P_e} \tag{3.16}$$

and

$$K_a = P_e^{|V|\times|V_\Gamma|}B_a. \tag{3.17}$$

Finally, replacing Equation (3.15) into Equation (3.4) the likelihood function is

$$P(G|\Gamma,\mathbf{S}) = \prod_{a\in V}\sum_{\alpha\in V_\Gamma} K_a \exp[\mu \sum_{b\in V}\sum_{\beta\in V_\Gamma} A_{ab}M_{\alpha\beta}s_{b\beta}]. \tag{3.18}$$

3.3 Learning the Supergraph

Having the *a posteriori* probability in hand, we proceed to measure the likelihood of the sample graphs. Let the graphs in the sample set be $\mathcal{G} = \{G_1,\ldots,G_i,\ldots,G_N\}$ and the supergraph be Γ. We use the set of assignment matrices $\mathcal{S} = \{\mathbf{S}^1,\ldots,\mathbf{S}^i,\ldots,\mathbf{S}^N\}$ to represent the correspondences between the nodes of sample graphs and those of the supergraph. Under the assumption that the graphs in $\mathcal{G}$ are independent samples from the distribution, the likelihood of the sample graphs can be written as follows using the *a posteriori* probabilities reviewed in Section 3.2

$$P(\mathcal{G}|\Gamma,\mathcal{S}) = \prod_{G_i\in\mathcal{G}}\prod_{a\in V_i}\sum_{\alpha\in V_\Gamma} K_a^i \exp[\mu \sum_{b\in V_i}\sum_{\beta\in V_\Gamma} A_{ab}^i M_{\alpha\beta}s_{b\beta}^i]. \tag{3.19}$$

We aim to locate the supergraph that maximizes the likelihood function above which is a mixture model over the set of possible correspondences. In [61], Luo and Hancock use this probabilistic model to cast the problem of graph matching into that of seeking the assignment matrix that maximizes this *a posterior* probability. To solve this problem, they develop an EM algorithm in which the node correspondences can be efficiently recovered using singular value decomposition. In our method we use this *a posterior* probability as the probability distribution of the sample graphs given the supergraph and correspondence information. However, to maximize the likelihood of the sample graphs we need to estimate not only the assignment matrices, but also the structure of the supergraph. In order to deal with the missing node assignment matrices and the structure of the supergraph, we develop a different EM algorithm to locate the solution.

3.3.1 Expected Log-Likelihood Function

We proceed to compute the expected value of the log-likelihood function of the sample graphs. The likelihood function for observing a sample graph G, i.e. for it to be generated by the supergraph Γ, is the *a posterior* probability in Equation (3.18), its log-likelihood function is

$$\mathcal{L}(\mathbf{S}) = \sum_{a \in V} \ln\{\sum_{\alpha \in V_\Gamma} K_a \exp[\mu \sum_{b \in V} \sum_{\beta \in V_\Gamma} A_{ab} M_{\alpha\beta} s_{b\beta}]\}. \tag{3.20}$$

According to [26] [78] [14] [61], Luo and Hancock show that the expectation of this log-likelihood function is

$$\Lambda(\mathbf{S}^{(n+1)}|\mathbf{S}^{(n)}) = \sum_{a \in V} \sum_{\alpha \in V_\Gamma} Q_{a\alpha}^{(n)} \{\ln\ K_a + \mu \sum_{b \in V} \sum_{\beta \in V_\Gamma} A_{ab} M_{\alpha\beta} s_{b\beta}^{(n+1)}\}, \tag{3.21}$$

where $\mathbf{Q}^{(n)}$ is a matrix with elements $Q_{a\alpha}^{(n)}$ that are equal to the *a posteriori* probability of node a in G being matched to node α in Γ at iteration *n* of the EM algorithm.

To develop the expected log-likelihood function for our supergraph model, since we do not know the supergraph adjacency matrix $\mathbf{M}$, we work with its expectation value $\mathbf{P}$. From the set of sample graphs with correspondence matrices represented by $\mathcal{S}$ we have

$$\bar{\Lambda}(\mathcal{S}^{(n+1)}|\mathcal{S}^{(n)}) = \sum_{G_i \in \mathcal{G}} \sum_{a \in V_i} \sum_{\alpha \in V_\Gamma} Q_{a\alpha}^{i,(n)} \{\ln K_a^i + \mu \sum_{b \in V_i} \sum_{\beta \in V_\Gamma} A_{ab}^i P_{\alpha\beta}^{(n)} s_{b\beta}^{i,(n+1)}\}, \quad (3.22)$$

where $P_{\alpha\beta}^{(n)} = E[\, M_{\alpha\beta}] = P(\, M_{\alpha\beta} = 1 |\, \mathcal{G}, \mathcal{S}^{(n)})$. Posed in this way, the estimation of the expectation value $P_{\alpha\beta}^{(n)}$ involves exploring all the configurations of the supergraph model, which is only computationally tractable using Monte Carlo sampling. The alternative is to assume a simple distribution for the supergraph edges. For instance, if we assume that the sample graph edges arise as independent samples from those of the supergraph under a Bernoulli distribution, then the likelihood becomes

$$P(\mathcal{G}|\Gamma, \mathcal{S}) = \prod_{G_i \in \mathcal{G}} \prod_{\alpha,\beta \in V_\Gamma} P_{\alpha\beta}^{\sum_{a,b \in V_i} s_{a\alpha}^i s_{b\beta}^i A_{ab}^i} (1 - P_{\alpha\beta})^{1 - \sum_{a,b \in V_i} s_{a\alpha}^i s_{b\beta}^i A_{ab}^i}. \quad (3.23)$$

This is a different distribution from the one prosed for matching but it is tractable. The trial success probability for the Bernoulli distribution $P_{\alpha\beta}$ is equal to the expected number of successes, and so

$$P_{\alpha\beta} = \frac{1}{|\mathcal{G}|} \sum_{G_i \in \mathcal{G}} \sum_{a,b \in V_i} s_{a\alpha}^i s_{b\beta}^i A_{ab}^i, \quad (3.24)$$

where $|\mathcal{G}|$ is the number of graphs in the sample set $\mathcal{G}$.

To maximize the expected log-likelihood function in Equation (3.22), since the first term under the curly braces contributes a constant amount

$$\sum_{G_i \in \mathcal{G}} \sum_{a \in V_i} \sum_{\alpha \in V_\Gamma} Q_{a\alpha}^{i,(n)} \ln K_a^i = \sum_{G_i \in \mathcal{G}} \sum_{a \in V_i} \ln K_a^i, \quad (3.25)$$

we confine our attention to the second term under the curly braces, which determines the update direction. The quantity of interest can be written as the summation of the traces of

products of matrices that is

$$\hat{\Lambda}(\mathcal{S}^{(n+1)}|\mathcal{S}^{(n)}) = \sum_{G_i \in \mathcal{G}} \mathrm{Tr}[(\mathbf{A}^i)^T \mathbf{Q}^{i,(n)} \mathbf{P}^{(n)} (\mathbf{S}^{i,(n+1)})^T]. \tag{3.26}$$

As a result, we concentrate on the critical quantity in Equation (3.26) and maximize its value.

3.3.2 Maximization

The maximization step involves recovering the elements in the assignment matrices $\mathcal{S}^{(n+1)}$ that satisfy the condition

$$\mathbf{S}^{i,(n+1)} = \arg\max_{\hat{\mathbf{S}}} \mathrm{Tr}[(\mathbf{A}^i)^T \mathbf{Q}^{i,(n)} \mathbf{P}^{(n)} \hat{\mathbf{S}}^T]. \tag{3.27}$$

To update those set of correspondence indicators, we use the extreme principal reported by Scott and Longuet-Higgins [89]. Scott and Longuet-Higgins demonstrate that the $\mathbf{S}^{i,(n+1)}$ satisfying the above condition can be recovered by performing the singular value decomposition

$$(\mathbf{A}^i)^T \mathbf{Q}^{i,(n)} \mathbf{P}^{(n)} = \mathbf{Y} \mathbf{\Sigma} \mathbf{U}^T, \tag{3.28}$$

where $\mathbf{Y}$ and $\mathbf{U}$ are orthogonal matrices and $\mathbf{\Sigma}$ is a rectangular diagonal matrix. From the factorization, we construct the matrix $\mathbf{\Delta}$ by making the diagonal elements in $\mathbf{\Sigma}$ unity, and compute matrix $\mathbf{Z}$ by setting $\mathbf{Z} = \mathbf{Y} \mathbf{\Delta} \mathbf{U}^T$. The elements of $\mathbf{Z}$ can be used to update the assignment indicators. However, the matrix $\mathbf{Z}$ is not a binary matrix in nature and the elements of $\mathbf{Z}$ are neither positive nor normalized. To overcome those problems, Scott and Longuet-Higggins suggest testing the elements of $\mathbf{Z}$ and transforming the matrix to a matrix of binary correspondence indicators. We follow their method and make the following setting. If the element $Z_{a\alpha}$ is the maximum value in both its containing row and

column, then the corresponding assignment indicator is set to unity; otherwise, it is set to zero. In other words,

$$S_{a\alpha}^{i,(n+1)} = \begin{cases} 1 & \text{if } Z_{a\alpha} = \max Z_{a\cdot} = \max Z_{\cdot\alpha} \\ 0 & \text{otherwise.} \end{cases} \tag{3.29}$$

There are alternative methods to update the assignment matrix $\mathbf{S}^{i,(n+1)}$ in this step. For instance, the graduated assignment method proposed by Gold and Rangarajan [48] could be adopted. Here we choose to use the idea of singular value decomposition for the reasons of simplicity.

3.3.3 Expectation

In the expectation step of the EM algorithm, we compute the matrix $\mathbf{Q}^{i,(n+1)}$ whose elements are the *a posteriori* probability of the nodes in the supergraph being matched to those of the sample graph G_i under the current correspondence $\mathbf{S}^{i,(n)}$. In [61], the *a posteriori* probability of a node in the supergraph graph Γ given a node in the sample graph G and the correspondence at iteration n is

$$P(\alpha|a, \mathbf{S}^{(n+1)}) = \frac{p(a|\alpha, \mathbf{S}^{(n)})\pi_\alpha^{(n)}}{\sum_{\alpha \in V_\Gamma} p(a|\alpha, \mathbf{S}^{(n)})\pi_\alpha^{(n)}} , \tag{3.30}$$

where

$$\pi_\alpha^{(n)} = \frac{1}{|V|} \sum_{a \in V} P(\alpha|a, \mathbf{S}^{(n)}). \tag{3.31}$$

Recall in Equation (3.15) we have

$$P(a|\alpha, \mathbf{S}) = K_a \exp[\mu \sum_{b \in V} \sum_{\beta \in V_\Gamma} A_{ab} M_{\alpha\beta} s_{b\beta}].$$

Replacing it into Equation (3.30), the *a posteriori* probability of the nodes in the supergraph graph Γ at iteration $n+1$ is

$$Q_{a\alpha}^{i,(n+1)} = \frac{\exp[\sum_{b\in V_i}\sum_{\beta\in V_\Gamma} A_{ab}^i P_{\alpha\beta}^{(n)} s_{b\beta}^{i,(n)}]\pi_\alpha^{i,(n)}}{\sum_{\alpha'\in V_\Gamma}\exp[\sum_{b\in V_i}\sum_{\beta\in V_\Gamma} A_{ab}^i P_{\alpha\beta}^{(n)} s_{b\beta}^{i,(n)}]\pi_{\alpha'}^{i,(n)}} \tag{3.32}$$

where

$$\pi_{\alpha'}^{i,(n)} = \frac{1}{|V_i|}\sum_{a\in V_i} Q_{a\alpha'}^{i,(n)}. \tag{3.33}$$

To run the EM algorithm, we need to initialize both the structure of the supergraph and the node correspondences between the sample graphs and the supergraph. Initializing the supergraph with different structure, the supergraph we learned using the EM algorithm could be different. The initial supergraph should have two properties. First, the initial supergraph should be easily obtained, and second it should preserve enough structural variations of the graphs in the sample set. Later in the experimental part, we will show how we construct a concatenated graph that satisfies the above properties and use it as the initial supergraph of the EM algorithm.

3.4 Experiments

In this section, we test our proposed method on a real-world "toys" dataset and provide some experimental evaluations of our generative model. The dataset used consists of images of 4 objects, with 20 different views of each object. We extract feature points in the images using the SIFT [60] detector and construct the sample graphs using Delaunay triangulation of the detected points. In Figure 3.1, we illustrate some example images of the objects and the extracted SIFT feature points on the images. Figure 3.2 shows the associated Delaunay graphs constructed from the SIFT points.

To initialize the structure of the supergraph, we construct a concatenated graph. The concatenated graph is constructed using the following procedures. we first match pairs of neighbour graphs from the same object using the SIFT feature descriptors and then merge the common structures for pairs of graphs. Finally we concatenate the common

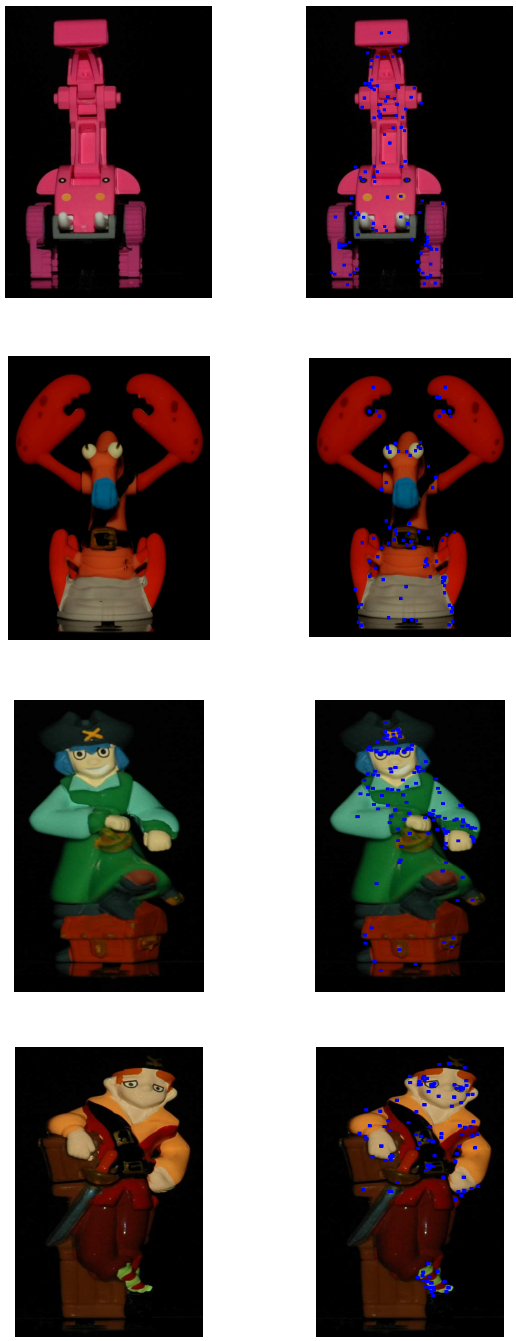


Figure 3.1: Example images and the extracted SIFT feature points on the images.

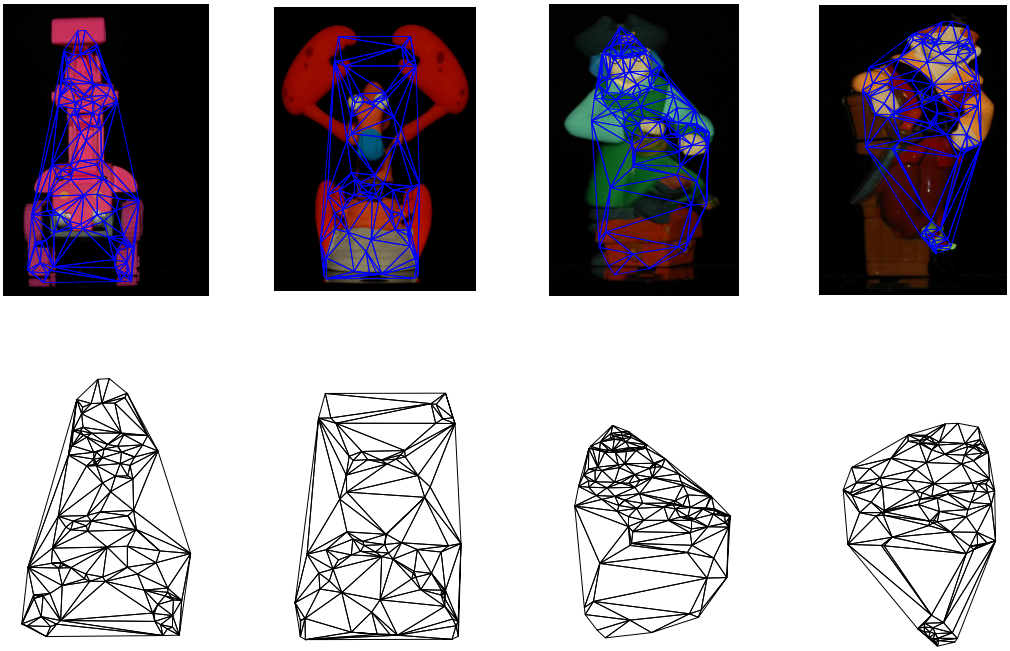


Figure 3.2: Example images and their associated graphs from the SIFT feature points.

structures over for the sample graphs to form the concatenated graph. The concatenated graph constructed in this way well preserves the structural variations present in the set of sample graphs.

The first part of our experimental investigation aims to validate the supergraph learning method. We iterate the two steps of the EM algorithm 50 times, and observe how the structure of the supergraph changes and how the likelihood function changes with iteration number. During the iterations of the EM algorithm, we recover the structure of the supergraph at iteration *n* by setting

$$M_{\alpha\beta}^{(n)} = \begin{cases} 1 & \text{if } P_{\alpha\beta}^{(n)} > 0 \\ 0 & \text{otherwise,} \end{cases} \tag{3.34}$$

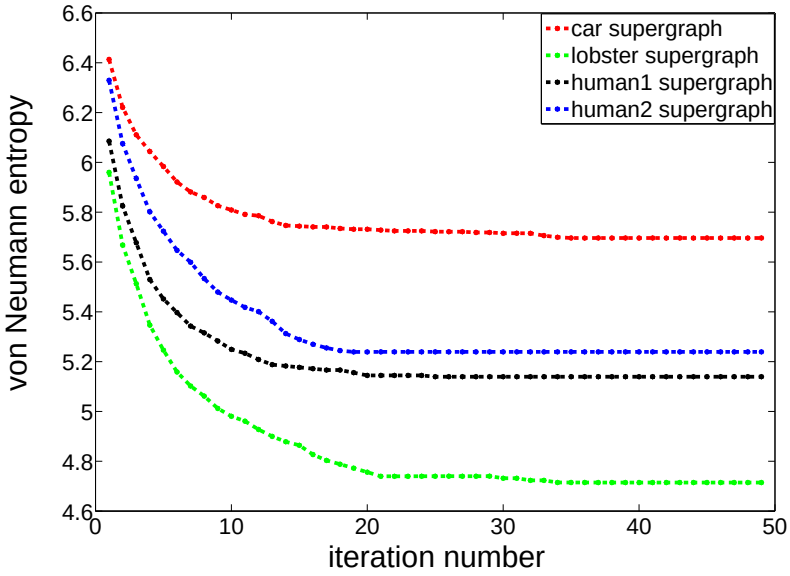
and measure the variation of the supergraph structure using the von Neumann entropy mentioned in Section 2.6.1. According to Passerini and Severini [70], the von Neumann

entropy of graphs is defined as

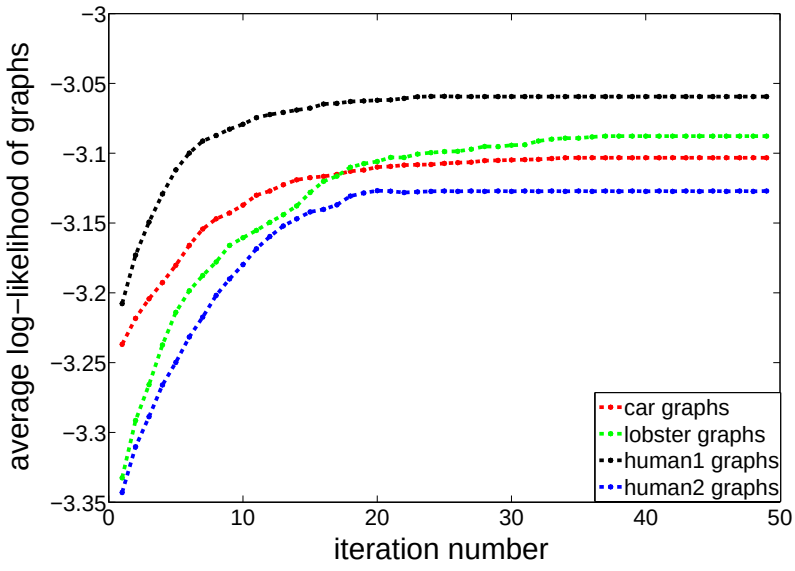
$$H(\Gamma) = -\sum_{j}^{|V_\Gamma|} \frac{\hat{\lambda}_j}{|V_\Gamma|} \ln \frac{\hat{\lambda}_j}{|V_\Gamma|}, \tag{3.35}$$

where $\hat{\lambda}_j$ are the eigenvalues of the normalized Laplacian matrix of the supergraph that is defined as $\hat{\mathbf{L}} = \mathbf{D}^{-1/2}(\mathbf{D} - \mathbf{M})\mathbf{D}^{-1/2}$, where $\mathbf{D}$ is the degree matrix of the supergraph which is a diagonal matrix with node degree on the diagonal line and $\mathbf{M}$ is the adjacency matrix. The von Neumann entropy can be used as an indicator of structural complexity of the supergraph. A detailed description of this entropy is provided in the following chapter. From Figure 3.3(a), it is clear that the von Neumann entropy of the supergraph decreases as the iteration number increases and finally converges when the iteration number increases to 40. This indicates that the complexity of the supergraph decreases and its structure becomes condensed and simplified as the number of iterations increases. Figure 3.3(b) shows that the average of the logarithm of the product of the *a posteriori* probabilities of the sample graphs, i.e. the average log-likelihood, increases and gradually converges as the number of iterations increases. In other words, our algorithm behaves in a stable manner both increasing the likelihood of sample graphs and simplifying the supergraph structure. Both the likelihood of sample graphs and the value of von Neumann entropy of the supergraph converge when the iteration number increases to 40.

Secondly, we evaluate the effectiveness of our generative model learned using the EM algorithm for classifying graphs. To do this, we learn a supergraph for each object class from a set of samples in the training set and use the learned supergraphs to distinguish graphs from a separate test set. For each graph in the test set, we compute its likelihood from a given supergraph using the *a posteriori* probability in Equation (3.18). The class-label of the test graph is determined by the class of the supergraph which gives the maximum *a posteriori* probability. The classification rate is the fraction of correctly identified graphs in the test set computed using 10-fold cross validation. For comparison,



(a)



(b)

Figure 3.3: (a) variation of the von Neumann entropy of the supergraph in Equation (3.35) during iterations and (b) variation of the average log-likelihood of the sample graphs during iterations.

Table 3.1: Comparison of the classification results. The values are the average classification rates from 10-fold cross validation, followed by their standard error. The highest classification rates are shown in bold.

Supergraph construction	classification rate
initial supergraph	$66.3\% \pm 0.038$
set median graph	$65.5\% \pm 0.025$
learned supergraph	$\mathbf{72.5\% \pm 0.022}$

we have also investigated the results obtained using two alternative constructions of the supergraph. The first of these is the initial structure concatenated from the results of SIFT descriptors. The second is the set median graph [56], i.e. the set median graph is a sample graph in the training set that has largest average value of the *a posteriori* probabilities to the other sample graphs in the training set. Table 3.1 shows the classification results obtained with the three different supergraph constructions. Among the three constructions, our learned supergraph achieves an average classification rate of 72.5%, which is higher than the initial supergraph's classification rate (66.3%) and the set median graph's (65.5%).

Finally, we visualize the structure of the learned supergraph for car object after the EM iterations in Figure 3.4.

3.5 Conclusions

Our first contribution of this chapter is that we have proposed a method of learning a generative model or supergraph for graphs. We began by introducing the *a posteriori* probability defined in a graph matching problem [61]. In the subsequent development, we used this probability to measure the likelihood of a sample graph from the supergraph.

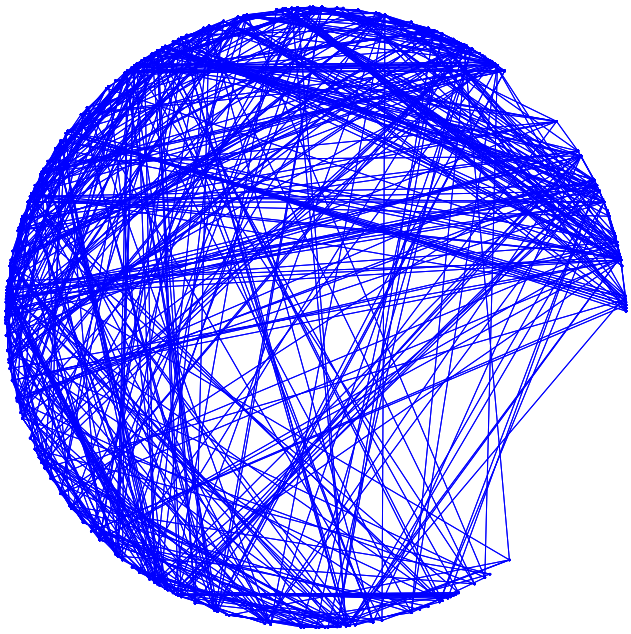


Figure 3.4: Learned supergraph for car object after the EM algorithm.

The supergraph we aim to learn is one which maximizes the likelihood of the sample graphs. Our second contribution is that, to maximize this objective function, we have developed an EM algorithm to maximize the likelihood of the sample graph and locate the structure of the optimal supergraph. In our experiments, we have demonstrated that our supergraph learning method can locate the structure of a supergraph that is optimal or suboptimal and have shown that the supergraph learned is effective for classification. Besides, we also have investigated the use of the von Neumann entropy as the indicator for measuring the complexity of the supergraph in the experimental part of this chapter.

Chapter 4

Graph Characterizations From Von Neumann Entropy

4.1 Introduction

In this chapter we explore how the von Neumann entropy can be used as a measure of graph characterization. We also develop a simplified form for the von Neumann entropy of a graph that can be computed in terms of node degree statistics. We compare the resulting characterizations with a number of different graph characterizations including Estrada's heterogeneity index [37] and the derivative of the Riemann zeta function at the origin [111]. In the case of Estrada's heterogeneity index we reveal a new link between Estrada's index and the commute time on a graph. We then proceed to show how the the von Neumann entropy can be used to compute thermodynamic depth and illustrate its applications to a set of protein-protein interaction networks.

The main drawback of randomness complexity is that it does not capture properly the correlations between vertices [39]. Statistical complexity aims to overcome this problem by measuring irregularities beyond randomness, and does not necessarily grow monotonically with randomness. Here we take the view that a more natural route to computing

graph complexity is to turn to information theory and to use entropy measures.

The novel contributions of this chapter are threefold. First, we develop new graph characterizations from the von Neumann entropy. Second, we reveal a new link between Estradas index and the commute time on a graph. Third, we show how to use the von Neumann entropy to construct the Bregman balls needed to compute the entropy based thermodynamic depth complexity. The outline of the chapter is as follows. In Section 4.2 we introduce the definition of the von Neumann entropy and show how to simplify and approximate its calculation. Section 4.3 describes the heterogeneity index and reveals its link to the commute time. Section 4.4 we review the derivative of the Riemann zeta function at the origin as an alternative graph characterization for experimental comparison. Section 4.5 describes the thermodynamic depth complexity measure for graphs, and explains how our von Neumann entropy can lead to the von Neumann entropy based thermodynamic depth complexity. Section 4.6 provides experimental results. This study is divided into three parts, namely a) an investigation of the relationship between the von Neumann entropy and its approximate counterpart, b) the comparison with alternative graph characterizations and c) the application of the entropy-based thermodynamic depth to protein-protein interaction networks. Section 4.7 offers some conclusions.

4.2 Graph Representation and the Von Neumann Entropy

The von Neumann entropy was originally defined in quantum mechanics as the Shannon entropy associated with the eigenvalues of the density matrix. Recently, Severini *et al.* [2] [70] have shown how to apply the von Neumann entropy to the domain of graphs through a mapping between discrete Laplacians and quantum states [16]. In the graph domain, the von Neumann entropy is the entropy of the density matrix obtained by scaling the normalized discrete Laplacian matrix by the reciprocal of the size of the graph. In the following we show how we derive this entropy.

To be consistent with the notations in Chapter 3, we denote the data graph under study by $G = (V, E)$, where V is the set of nodes and E is the set of edges. Further, the structure of the graph is represented by a $|V| \times |V|$ adjacency matrix $\mathbf{A}$ ($|V|$ is the number of the nodes in the graph) whose elements are

$$A_{ab} = \begin{cases} 1 & \text{if } (a,b) \in E \\ 0 & \text{otherwise.} \end{cases} \tag{4.1}$$

The degree matrix of graph G is a diagonal matrix $\mathbf{D}$, whose diagonal elements are given by $D_{aa} = d_a = \sum_{b \in V} A(a,b)$. From the degree matrix and the adjacency matrix we can construct the Laplacian matrix $\mathbf{L} = \mathbf{D} - \mathbf{A}$, i.e. the degree matrix minus the adjacency matrix. The elements of the Laplacian matrix are

$$L_{ab} = \begin{cases} d_a & \text{if } a = b \\ -1 & \text{if } (a,b) \in E \\ 0 & \text{otherwise.} \end{cases} \tag{4.2}$$

The normalized Laplacian matrix is given by $\hat{\mathbf{L}} = \mathbf{D}^{-1/2}\mathbf{L}\mathbf{D}^{-1/2}$ and has elements

$$\hat{L}_{ab} = \begin{cases} 1 & \text{if } a = b \text{ and } d_a \neq 0 \\ -\frac{1}{\sqrt{d_a d_b}} & \text{if } (a,b) \in E \\ 0 & \text{otherwise.} \end{cases} \tag{4.3}$$

The spectral decomposition of the normalized Laplacian matrix is $\hat{\mathbf{L}} = \hat{\mathbf{\Phi}}\hat{\mathbf{\Lambda}}\hat{\mathbf{\Phi}}^T$ where $\hat{\mathbf{\Lambda}} = \text{diag}(\hat{\lambda}_1, \hat{\lambda}_2, ..., \hat{\lambda}_{|V|})$ is a diagonal matrix with the ordered eigenvalues as elements $(0 = \hat{\lambda}_1 < \hat{\lambda}_2 < ... < \hat{\lambda}_{|V|})$ and $\hat{\mathbf{\Phi}} = (\hat{\phi}_1|\hat{\phi}_2|...|\hat{\phi}_{|V|})$ is a matrix with the corresponding ordered orthonormal eigenvectors as columns. The normalized Laplacian matrix is positive semi-definite and so has all eigenvalues non-negative. The number of zero eigenvalues is the number of connected components in the graph. For a connected graph, there is only one eigenvalue which is equal to zero. The normalization factor means that

the largest eigenvalue is less than or equal to 2, with equality only when G is bipartite. Hence all the eigenvalues of the normalized Laplacian matrix are in the interval $[0, 2]$. The normalized Laplacian matrix is commonly used as a graph representation and the eigenvector $\hat{\phi}_2$ associated with the smallest non-zero eigenvalues $\hat{\lambda}_2$ is often used in graph cuts [83][92].

The trace of the normalized Laplacian matrix is equal to the size of the graph, i.e. the number of the nodes in the graph. Scaling the normalized Laplacian matrix by the reciprocal of its trace, we obtain a density matrix $\frac{\hat{\mathbf{L}}}{|V|}$. The eigenvalues of the density matrix is $(\frac{\hat{\lambda}_1}{|V|}, \frac{\hat{\lambda}_2}{|V|}, ..., \frac{\hat{\lambda}_{|V|}}{|V|})$ and thus the von Neumann entropy of density matrix associated with the normalized Laplacian matrix of the graph is defined as [70]

$$H(G) = -\sum_{j=1}^{|V|} \frac{\hat{\lambda}_j}{|V|} \ln \frac{\hat{\lambda}_j}{|V|} \,, \tag{4.4}$$

where $0 \ln 0 = 0$, by convention. The von Neumann entropy above relies on the computation of the normalized Laplacian spectrum, therefore its computational complexity is cubic in the number of nodes. To render the computation more efficient, we explore how to simplify and approximate the calculation of von Neumann entropy. The Taylor expansion for $\ln \frac{\hat{\lambda}_j}{|V|}$ at point 1 is

$$(\frac{\hat{\lambda}_j}{|V|} - 1) - \frac{1}{2}(\frac{\hat{\lambda}_j}{|V|} - 1)^2 + \frac{1}{3}(\frac{\hat{\lambda}_j}{|V|} - 1)^3 - \frac{1}{4}(\frac{\hat{\lambda}_j}{|V|} - 1)^4 + \cdots \,. \tag{4.5}$$

If we keep the first item of the expansion and discard the remaining that contribute to a small amount, $\ln \frac{\hat{\lambda}_j}{|V|}$ is approximated using $(\frac{\hat{\lambda}_j}{|V|} - 1)$. Then the entropy $-\sum_j \frac{\hat{\lambda}_j}{|V|} \ln \frac{\hat{\lambda}_j}{|V|}$ can be replaced by the quadratic entropy $\sum_j \frac{\hat{\lambda}_j}{|V|}(1 - \frac{\hat{\lambda}_j}{|V|})$, then we obtain

$$H(G) = -\sum_j \frac{\hat{\lambda}_j}{|V|} \ln \frac{\hat{\lambda}_j}{|V|} \simeq \sum_j \frac{\hat{\lambda}_j}{|V|}(1 - \frac{\hat{\lambda}_j}{|V|}) = \frac{1}{|V|}\sum_j \lambda_j - \frac{1}{|V|^2}\sum_j \lambda_j^2 \,. \tag{4.6}$$

Using the fact that $\mathrm{Tr}[\hat{\mathbf{L}}^k] = \sum_j \hat{\lambda}_j^k$ [12], the quadratic entropy can be rewritten as

$$\bar{H}(G) = \frac{\mathrm{Tr}[\hat{\mathbf{L}}]}{|V|} - \frac{\mathrm{Tr}[\hat{\mathbf{L}}^2]}{|V|^2} \,. \tag{4.7}$$

According to Equation (4.3), the normalized Laplacian matrix $\hat{\mathbf{L}}$ has unit diagonal elements, therefore for the trace of the normalized Laplacian matrix we have

$$\mathrm{Tr}[\hat{\mathbf{L}}] = |V|. \tag{4.8}$$

Similarly, for the trace of the square of the normalized Laplacian, we have

$$\begin{aligned}
\mathrm{Tr}[\hat{\mathbf{L}}^2] &= \sum_{a\in V}\sum_{b\in V} \hat{L}_{ab}\hat{L}_{ab} = \sum_{a\in V}\sum_{b\in V} (\hat{L}_{ab})^2 \\
&= \sum_{\substack{a,b\in V \\ a=b}} (\hat{L}_{ab})^2 + \sum_{\substack{a,b\in V \\ a\neq b}} (\hat{L}_{ab})^2 \\
&= |V| + \sum_{(a,b)\in E} \frac{1}{d_a d_b} \,.
\end{aligned} \tag{4.9}$$

Substituting Equation (4.8) and Equation (4.9) into Equation (4.7), the entropy becomes

$$\bar{H}(G) = \frac{\mathrm{Tr}[\hat{\mathbf{L}}]}{|V|} - \frac{\mathrm{Tr}[\hat{\mathbf{L}}^2]}{|V|^2} = \frac{|V|}{|V|} - \frac{|V|}{|V|^2} - \sum_{(a,b)\in E} \frac{1}{|V|^2\, d_a d_b} = 1 - \frac{1}{|V|} - \sum_{(a,b)\in E} \frac{1}{|V|^2\, d_a d_b} \,. \tag{4.10}$$

As a result, we can approximate the von Neumann entropy using two measures of graph structure. The first is the number of nodes of the graph, while the second is based on degree statistics for pairs of nodes connected by edges. The approximation can be computed without evaluating the spectrum of the normalized adjacency matrix (which is cubic). The expression of the approximate entropy is quadratic in the number of nodes in a graph.

4.3 Graph Heterogeneity Index

To compare our derived graph characterization with its alternatives, we review the network heterogeneity index recently developed by Estrada [37] and reveal its link to the commute time on a graph. To develop the heterogeneity index, Estrada commences by defining a local index which measures the irregularity of an edge in the graph $(a, b) \in E$ as

$$\delta_{ab} = [f(d_a) - f(d_b)]^2, \tag{4.11}$$

where $f(d_a)$ is a function of the node degree. Selecting $f(d_a) = d_a^{-1/2}$, the heterogeneity index proposed is defined to be the sum of the irregularity of all edges in the graph,

$$J'(G) = \sum_{(a,b)\in E} (d_a^{-1/2} - d_b^{-1/2})^2. \tag{4.12}$$

The main advantage of defining the index as the sum of square differences of a function of node degree is that the index can be expressed in terms of a quadratic form of the Laplacian matrix of the graph. That is, let $\mathbf{d}^{-1/2} = (d_1^{-1/2}, d_2^{-1/2}, ..., d_{|V|}^{-1/2})^T$ represent a column vector where d_a is the degree of the node a, the index can be written as

$$J'(G) = \sum_{(a,b)\in E} (d_a^{-1/2} - d_b^{-1/2})^2 = \frac{1}{2}(\mathbf{d}^{-1/2})^T \mathbf{L}\mathbf{d}^{-1/2}. \tag{4.13}$$

The index above can also be stated in terms of the *Randić index* $''\ {}^1R_{-1/2}\ ''$ [75] of the graph,

$$J'(G) = \sum_{(a,b)\in E} (d_a^{-1/2} - d_b^{-1/2})^2 = |V| - 2\sum_{(a,b)\in E} (d_a d_b)^{-1/2} = |V| - 2\ {}^1R_{-1/2}. \tag{4.14}$$

Li and Shi [58] show that for connected graphs the *Randić index* is bounded as follows

$$\sqrt{|V| - 1} \leq {}^1R_{-1/2} \leq \frac{|V|}{2}, \tag{4.15}$$

where the lower bound is attained for star graphs and the upper bound is attained for regular graphs with $|V|$ nodes. Thus the heterogeneity index is bounded as follows

$$0 \leq J'(G) = |V| - 2\,{}^1R_{-1/2} \leq |V| - 2\sqrt{|V|-1}\,. \tag{4.16}$$

Then Estrada defines the normalized heterogeneity index as

$$J(G) = \frac{|V| - 2\,{}^1R_{-1/2}}{|V| - 2\sqrt{|V|-1}} = \frac{\sum_{(a,b)\in E}(d_a^{-1/2} - d_b^{-1/2})^2}{|V| - 2\sqrt{|V|-1}}$$

$$= \frac{1}{|V| - 2\sqrt{|V|-1}} \sum_{(a,b)\in E} (\frac{1}{d_a} + \frac{1}{d_b} - \frac{2}{\sqrt{d_a d_b}})\,. \tag{4.17}$$

The value of the normalized heterogeneity index is in the range $[0, 1]$, i.e. $0 \leq J(G) \leq 1$. It is zero for regular graphs and one for star graphs. Heterogeneous starlike graphs are expected to have values of $J(G)$ close to one. On the other hand, more regular graphs are expected to have values close to zero.

It is interesting to note that Maier *et al.* [65] have shown that $1/d_a + 1/d_b$ is proportional to the commute time CT_{ab} (or resistance distance) between nodes a and b for graphs of large degree. Therefore, in the limit of large node degree we have

$$J(G) \sim \sum_{(a,b)\in E} \{CT_{ab} - 2\hat{A}_{ab}\} \tag{4.18}$$

where $\hat{\mathbf{A}} = \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$ is the normalized adjacency matrix with elements $\hat{A}_{ab} = \frac{1}{\sqrt{d_a d_b}}$ when $(a, b) \in E$ and otherwise zero. The heterogeneity is largest when the commute time between nodes a and b differs from $2\hat{A}_{ab}$ due to a large number of alternative connecting paths.

Recall that commute time is the average of the outward hitting time and return hitting time, over all paths connecting a pair of nodes [74]. It hence provides a non-local index

of connectivity between pairs of nodes, which is non-zero even if there is no connecting edge. Apart from the commute time term and constants related to the size of the graph, the simplified von Neumann entropy depends on

$$\Psi_{\bar{H}(G)} = -\sum_{(a,b)\in E} \frac{1}{|V|^2\, d_a d_b} = -\frac{1}{|V|^2} \sum_{(a,b)\in E} \hat{L}_{ab}^2\,, \tag{4.19}$$

whereas the normalized heterogeneity index depends on

$$\Psi_{J(G)} = -\sum_{(a,b)\in E} \frac{2}{\sqrt{d_a d_b}} = 2 \sum_{(a,b)\in E} \hat{L}_{ab}. \tag{4.20}$$

Hence, the heterogeneity contains measures of both global path length distribution via commute time, and local edge structure via the elements of the normalized Laplacian. The entropy on the other hand is based only on the latter.

4.4 Riemann Zeta Function Derivative

In this section we review a unary representation based on the analysis of the Riemann zeta function which will be used for comparisons in the experimental part. The Riemann zeta function associated with normalized Laplacian eigenvalues is defined to be [111]

$$\zeta(\upsilon) = \sum_{\hat{\lambda}_j \neq 0} \hat{\lambda}_j^{-\upsilon}, \tag{4.21}$$

which is the result of exponentiating and summing the reciprocal of the non-zero normalized Laplacian eigenvalues.

The derivative of the zeta function is given by

$$\zeta'(\upsilon) = \sum_{\hat{\lambda}_j \neq 0} -\hat{\lambda}_j^{-\upsilon} \ln \hat{\lambda}_j. \tag{4.22}$$

At the origin the derivative takes on the value

$$\zeta'(0) = \sum_{\hat{\lambda}_j \neq 0} \{-\ln \hat{\lambda}_j\} = \ln\{\prod_{\hat{\lambda}_j \neq 0} \frac{1}{\hat{\lambda}_j}\}. \tag{4.23}$$

McKay [67] has shown that the derivative of the zeta function at the origin is linked to the number of spanning trees in a graph G through

$$\tau(G) = \frac{\prod_{a \in V} d_a}{\sum_{a \in V} d_a} \exp[-\zeta'(0)]. \tag{4.24}$$

As a result, the derivative of the Riemann zeta function at the origin is determined by the number of spanning trees in the graph together with the degree of its nodes.

4.5 Thermodynamic Depth Complexity

Escolano *et al.* [36] [35] have recently explored how to measure the complexity of graphs using thermodynamic depth. They consider the nodes in a graph as microscopic states and their expansion subgraphs as macroscopic states and in this way they define a node history. Given a graph $G = (V, E)$, then the history of a node $a \in V$ is $\hbar_a(G) = \{e(a), e^2(a), ..., e^q(a)\}$ where $e(a) \subseteq G$ is the first order expansion subgraph given by a and all $b : (a, b) \in E$, $e^2(a) = e(e(a)) \subseteq G$ is the second-order expansion consisting of $c : (b, c) \in E, b \in V_{e(a)}, c \notin V_{e(a)}$, and so on until q cannot be increased. If G is connected $e^q(a) = G$, otherwise $e^q(a)$ is the connected component to which a belongs.

Every node history $\hbar_a(G)$ specifies a different causal trajectory leading to G or its connected components. If the causal trajectories are confined with narrow bounds, then the graph G (or its connected components) is easy to reach. In this case the process leading to the graph and generating the trajectories is simple, and the thermodynamic depth of the graph is shallow. Otherwise if a wide range of historical alternatives has been extracted, then the process is complex and the graph has a deep thermodynamic depth.

In this section, we develop a novel variant of this idea and use the von Neumann

entropy of the expansion subgraph as a complexity characterization. Specifically, our characterization is developed based on the idea of Escolano *et al.* [36] [35], where they use the centres and radii of the minimum enclosing Bregman balls (MEBB) [69] to characterize the causal trajectory of each node of the graph. The Bregman divergence [17] is used in information theory to assess the similarity between two objects. Given the von Neumann entropies of two subgraphs (h_1 and h_2) and a strictly convex and differentiable function g on $\mathcal{X}$, the Bregman divergence associated with g for points h_1 and h_2 is

$$B(h_1 \parallel h_2) = g(h_1) - g(h_2) - (h_1 - h_2)\nabla g(h_2). \tag{4.25}$$

If we use $g(h) = h \ln h - h$, the distance becomes the Kullback-Leibler divergence

$$KL(h_1 \| h_2) = h_1 \ln \frac{h_1}{h_2} - h_1 + h_2. \tag{4.26}$$

We characterize the causal trajectory of a node by the centre and the radius of the smallest enclosing Bregman ball that encloses the entropy values of all expansion subgraphs for that node history. More specifically, given $\hbar_a(G)$, the von Neumann entropy $h_l = H(e^l(a))$ for the l-th expansion of a and Kullback-Leibler divergence KL, the casual trajectory leading to G (or one of its connected components) from a is characterized by the centre $c_a \in \mathbb{R}$ and radius $r_a \in \mathbb{R}$ of the MEBB $\mathcal{B}^{c_a,r_a} = \{h_l \in \mathcal{X} : KL(c_a \| h_l) \leq r_a\}$. Solving for the centre and radius implies finding c_a that minimize r_a subject to $KL(c_a \| h_l) \leq r_a,\ \forall\, l\ \ 1 \leq l \leq q$. Nock and Nielson [69] proposed an efficient algorithm to estimate the centre c_a by iterating

$$c_a^{(n)} \leftarrow \nabla_g^{-1}(\frac{n}{n+1}\nabla_g(c_a^{(n-1)}) + \frac{1}{n+1}\nabla_g(h^{(n)})), \tag{4.27}$$

where n is the iteration number and

$$h^{(n)} = \arg \max_{h' \in \{h_1, h_2, \ldots, h_q\}} KL(c_a^{(n-1)} \| h') \,. \tag{4.28}$$

If h_l $(1 \leq l \leq q)$ is chosen at least once during iterations, its Lagrange multiplier $\sigma_l > 0$, and the radius is simply chosen as

$$r_a = \max_{\sigma_l > 0} KL(c_a \| h_l). \tag{4.29}$$

After characterizing the causal trajectories of a graph, the thermodynamic depth complexity of the graph is defined as follows. Given $G = (V, E)$, with node number $|V|$ and all the $|V|$ pairs (c_a, r_a), the entropy-thermodynamic depth complexity of G is characterized by the MEBB $\mathcal{B}^{c^*,r^*} = \{c_a \in \mathcal{X} : KL(c^* \| c_a) \leq r^*\}$ and $\Theta_{min} = \min_{h \in \mathcal{B}^{c^*,r^*}} KL(h^\infty \| h)$, where h^∞ is the von Neumann entropy of the van der Waerden matrix. The van der Waerden matrix is a $|V| \times |V|$ matrix with all entries equal to $\frac{1}{|V|}$. Then the thermodynamic depth of the graph is given by $\mathcal{D}(G) = r^* \times \Theta_{min}$.

We have shown how to use the von Neumann entropy as basic complexity measure to construct the Bregman ball and derive the entropy-based thermodynamic depth complexity. In fact, the thermodynamic depth approach can be applied to any structural complexity measure. In our experiments, we will compare it with thermodynamic depth based on Estrada's heterogeneity index and thermodynamic depth based on the derivative of the zeta function at the origin. An advantage of the thermodynamic depth complexity measure is that it overcomes problems of cospectrality when the basic complexity measure is associated with spectra of graphs. This is because the thermodynamic depth complexity relies on all expansion subgraphs from each node, rather than the single structure of the whole graph alone. In addition, the thermodynamic depth complexity is independent of the graph size, which means graphs with a large number of nodes do not necessarily have a large complexity.

4.6 Experiments

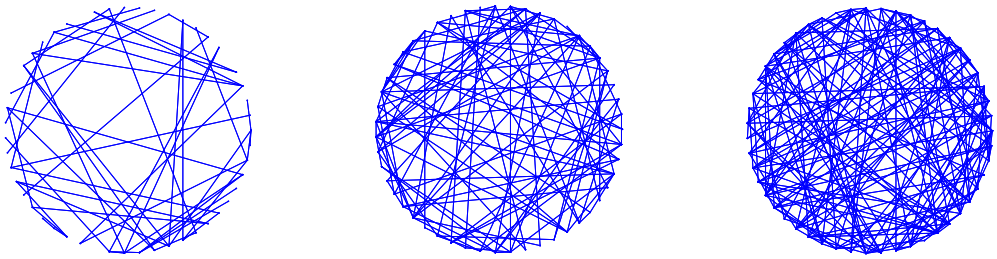
The experimental evaluation of the different graph characterizations is divided into three parts. We commence with a study on both the synthetic data and real world data which

aims to evaluate how well the approximation of the von Neumann entropy holds. The second part is concerned with a comparison of the use of the graph characterizations as a means of representing graph structure for the purpose of object recognition on the real-world data. In the third part we embed the von Neumann entropy into the thermodynamic depth approach and use the derived complexity measure to characterize sets of protein-protein interaction networks.

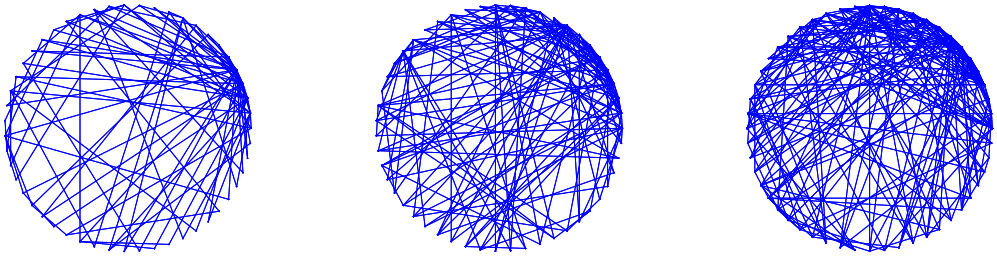
4.6.1 Approximation Evaluation

We first focus on analyzing how well the approximation of the von Neumann entropy holds. Recall that in Section 4.2 we show that we approximate the value of the von Neumann entropy of a graph using the number of nodes in the graph together with the node degree statistics. This approximation is realized by replacing the entropy $-\sum_j \frac{\hat{\lambda}_j}{|V|} \ln \frac{\hat{\lambda}_j}{|V|}$ by the quadratic entropy $\sum_j \frac{\hat{\lambda}_j}{|V|}(1 - \frac{\hat{\lambda}_j}{|V|})$. To explore how well the approximation holds, we experiment with both a synthetic graph dataset and Delaunay graphs from a real-world image dataset.

Synthetic dataset. The synthetic dataset contains two types of representative graph models. The first are the classical Erdös-Rényi (ER) random-graphs [33]. These are generated by connecting pairs of nodes in the graphs with an equal probability p ($0 \leq p \leq 1$). The second class of graphs are the Barabási-Albert (BA) scale-free graphs. Their degree distribution follows the power-law distribution shared by many real-world networks. The number of the nodes of the ER graphs varies from 50 to 70. For each number of nodes we generated several ER graphs with different values of p. The BA scale-free graphs here are generated with the preferential attachment algorithm in [7]. The preferential attachment commences from a small seed graph of size m_0 and iteratively introduces one new node to the graph by connecting it to m ($1 \leq m \leq m_0$) existing nodes with a probability that is proportional to the degrees of the existing nodes. We use a seed graph of size $m_0 = 5$ and different m values to generate BA graphs whose number of



(a) Synthetic ER graphs. Left: $p = 0.1$, middle: $p = 0.2$, right: $p = 0.4$.



(b) Synthetic BA graphs. Left: $m = 1$, middle: $m = 2$, right: $m = 3$.

Figure 4.1: Examples of synthetic graphs.

nodes ranges from 50 to 200. In Figure 4.1, we show some examples of the ER graphs and BA networks generated in this way.

Real-world dataset. The real-world image dataset used is the COIL dataset [68] which consists of images of different views of 3D objects, with 72 views of each object from equally spaced directions over 360°. We extract corner features using the corner detector [54] from each image and use the detected feature points as nodes to construct sample graphs by Delaunay triangulation. Some example images and their Delaunay graphs can be seen in Figure 4.2.

To investigate the veracity of the entropy approximation, we compute the von Neumann entropy of the three types of graphs together with their quadratic approximation. We also randomly select different sets of normalized eigenvalues from a uniform distri-

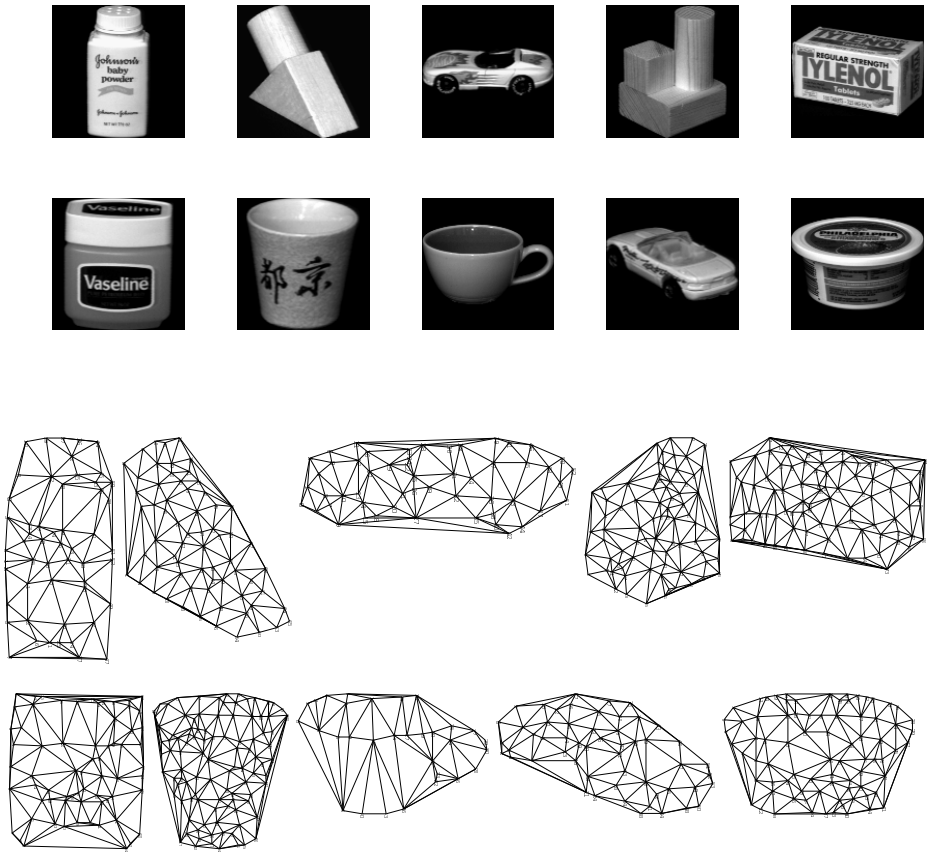
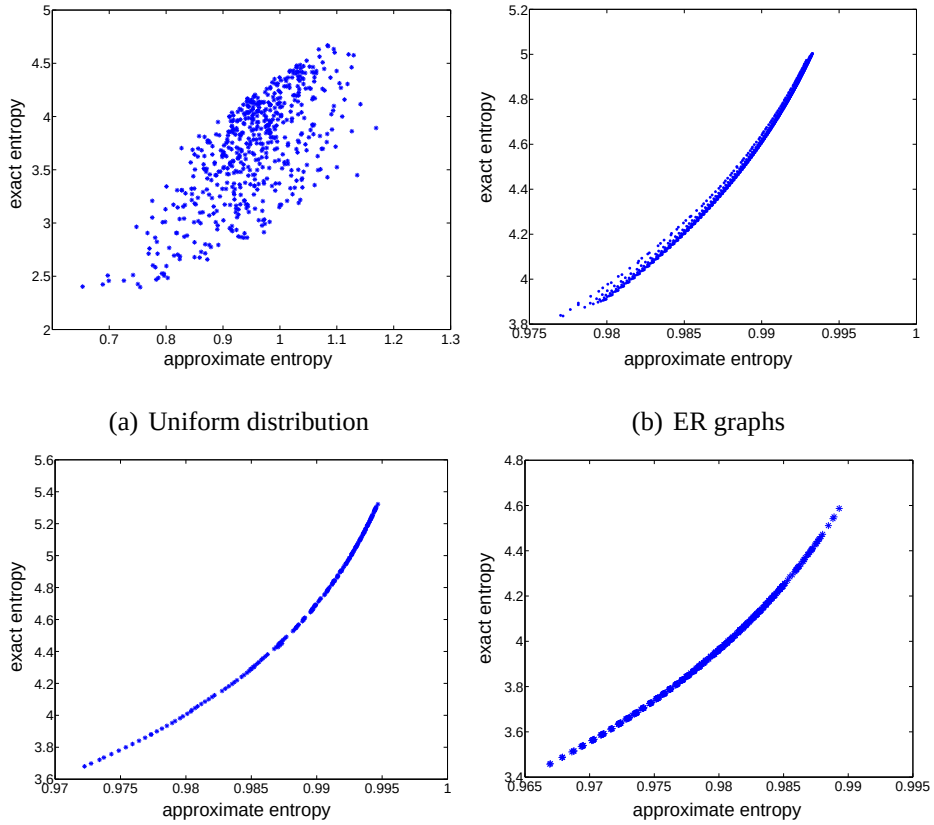


Figure 4.2: Example images from the COIL dataset and their associated Delaunay graphs.

bution between 0 and 2. For each set of the eigenvalues, we divide the eigenvalues by the number of the eigenvalues in the set, to ensure that the resulting values add up to one. We show the relationship between the exact von Neumann entropy computed from the resulting values and their approximate quadratic entropy. Figure 4.3 shows scatter plots of the von Neumann entropy (y-axis) versus the quadratic approximation (x-axis) for the uniform sample of eigenvalues and the three different types of graphs. Figure 4.3(a) shows the scatter plot for the uniform eigenvalue sample. Here the points disperse in a similar shape of an ellipse. Compared with the uniform eigenvalue sample, the scatter plot for the ER graphs in Figure 4.3(b) shows that the approximate entropy and the von Neumann



(a) Uniform distribution

(b) ER graphs

(c) BA graphs

(d) Delaunay graphs

Figure 4.3: Exact entropy versus approximate entropy for the synthetic dataset and COIL dataset.

entropy have a same variation pattern. They increase or decrease at the same time. The plot for the ER graphs has a small dispersion. For the BA graphs in Figure 4.3(c), there is again a same variation pattern between the two entropies, but less dispersion than the ER graphs. The scatter plot for the Delaunay graphs in Figure 4.3(d) demonstrates a similar result to that of the BA graphs. Note that the slope of the scatter plots for BA, ER and Delaunay graphs does not change dramatically, we may assume there is a linear dependence relationship between the approximate entropy and the von Neumann entropy when the exact computation is not strictly required. The same variation patterns or even linear regression trend for the three types of graphs indicates the approximate entropy of these

graphs is a good approximation. Recall that the computational complexity of obtaining the von Neumann entropy is governed by the spectral decomposition of the normalized Laplacian matrix. This requires $O(|V|^3)$ operations where $|V|$ is the number of nodes in a graph. On the other hand, the computational complexity of the approximate entropy is $O(|V|^2)$. Therefore, using the approximate entropy as a substitute for the von Neumann entropy offers an advantage of easy computation.

4.6.2 Comparison of Graph Characterizations

In this section, we turn our attention to comparing the utility of the two entropy measures with four alternative graph characterizations, i.e. the heterogeneity index, the derivative of the Riemann zeta function at the origin, average path length and graph diameter. To do this, we first select 5 objects from the COIL dataset and plot different characterization measures of their Delaunay graphs. From left-to-right and top-to-bottom in Figure 4.4 we show the values of six characterizations for different objects. In the plot, the x-axis is the object index and the y-axis is the value of the characterization. For each object there are 72 graphs extracted from images obtained with different viewpoints. The graphs from images of a same object are indicated by a same color. From Figure 4.4, we note that the four of the characterizations, i.e. the von Neumann entropy, the approximate entropy and the derivative of the zeta function at the origin and the average path length separate the objects well. On the other hand the values of the heterogeneity index and graph diameter overlap significantly for the different objects and do not distinguish the objects well.

To further quantitatively evaluate the use of the six methods on an object classification task, we apply a K-nearest neighbour classifier to the six graph characterizations of the Delaunay graphs for the objects in the COIL dataset. We observe how the classification rate changes as we increase the number of objects to be distinguished. Figure 4.5 shows the variation of the classification rates for the six graph characterizations. In our experiments, we set K=7 and the classification rate is the average fraction of graphs

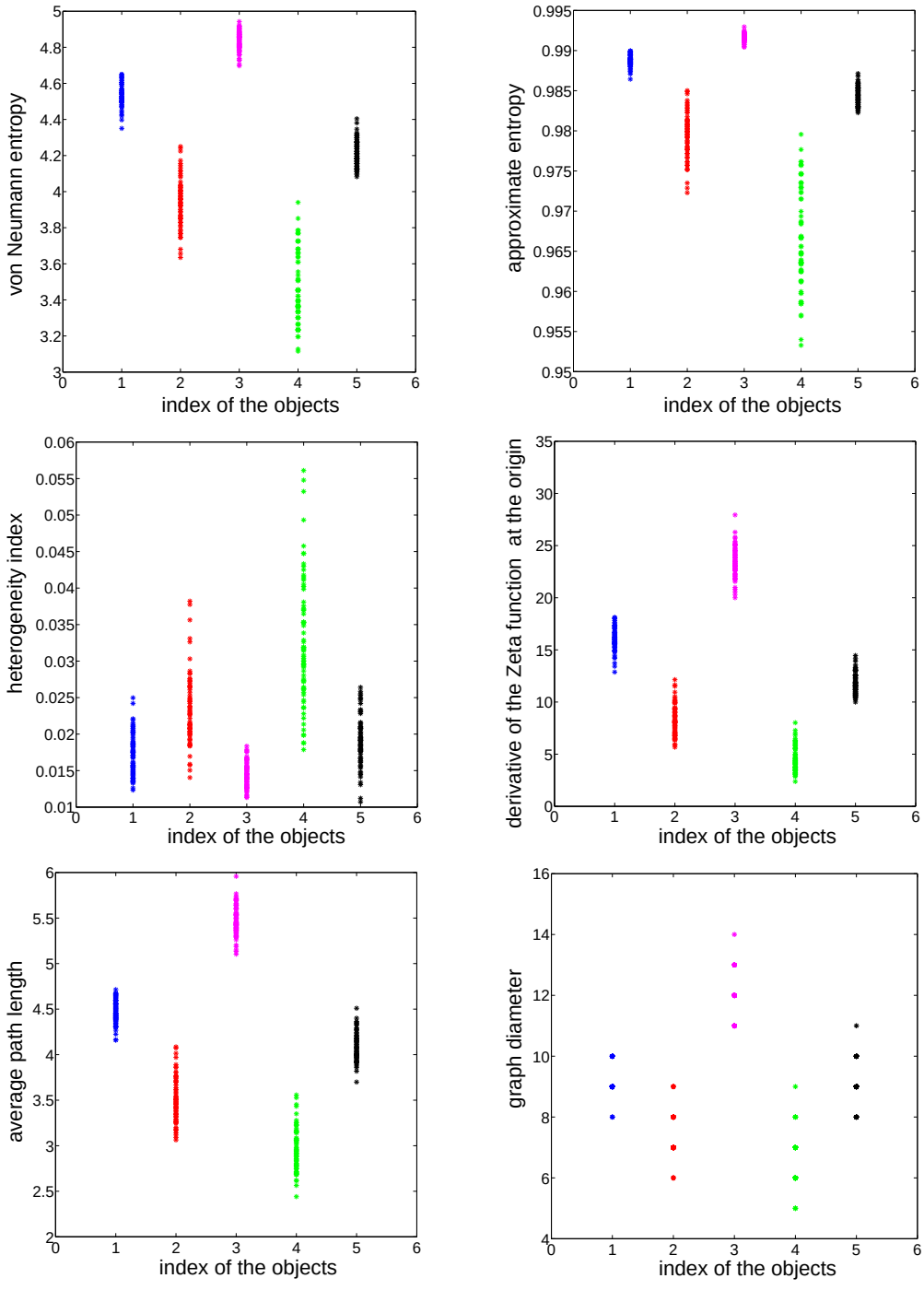


Figure 4.4: The values of alternative graph characterizations.

that are correctly identified, computed using 10-fold cross-validation. From the plot, it is clear that the von Neuman entropy method (red line) and the approximate entropy method (blue line) give almost the same results and they always achieve the highest classification rate as the number of objects increases from 5 to 15. The derivative of the zeta function at the origin (black line) follows the performance of the entropy methods. The average path length (cyan line) outperforms the graph diameter (green line) and the heterogeneity index (magenta line) has lowest classification rates on all the classification tasks.

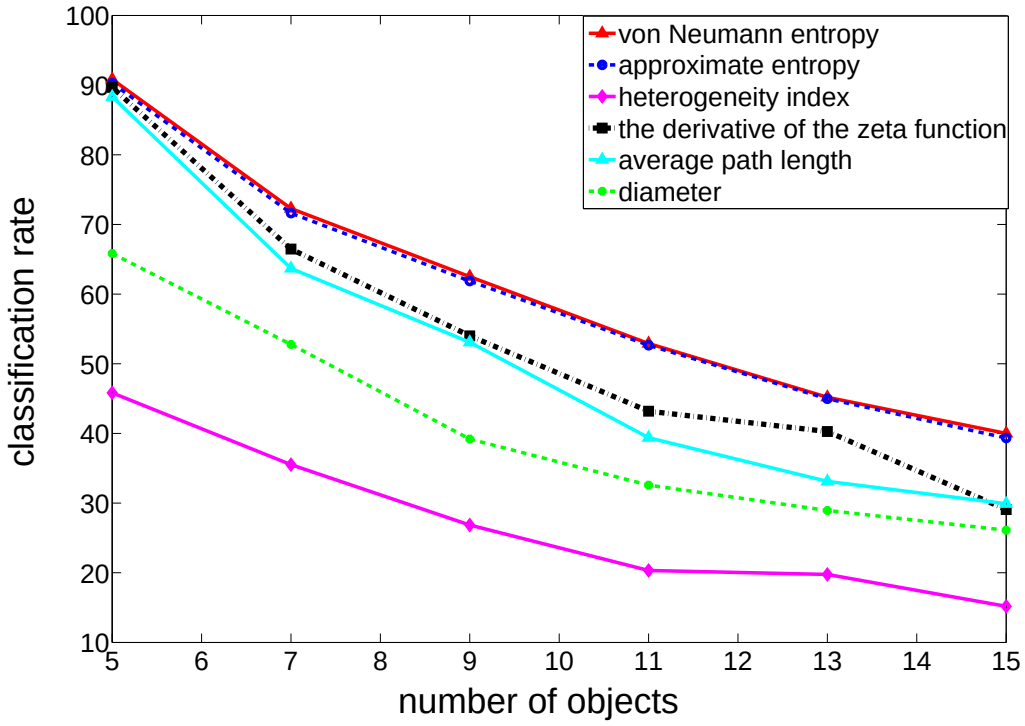


Figure 4.5: Comparison of the classification rate for the six methods.

4.6.3 Von Neumann Entropy Based Thermodynamic Depth

Having compared the graph characterizations, we apply the entropy-based thermodynamic depth complexity measures to analyze a set of protein-protein interaction networks (PPIs) [36]. Our aim in this experiment is to investigate whether the von Neumann en-

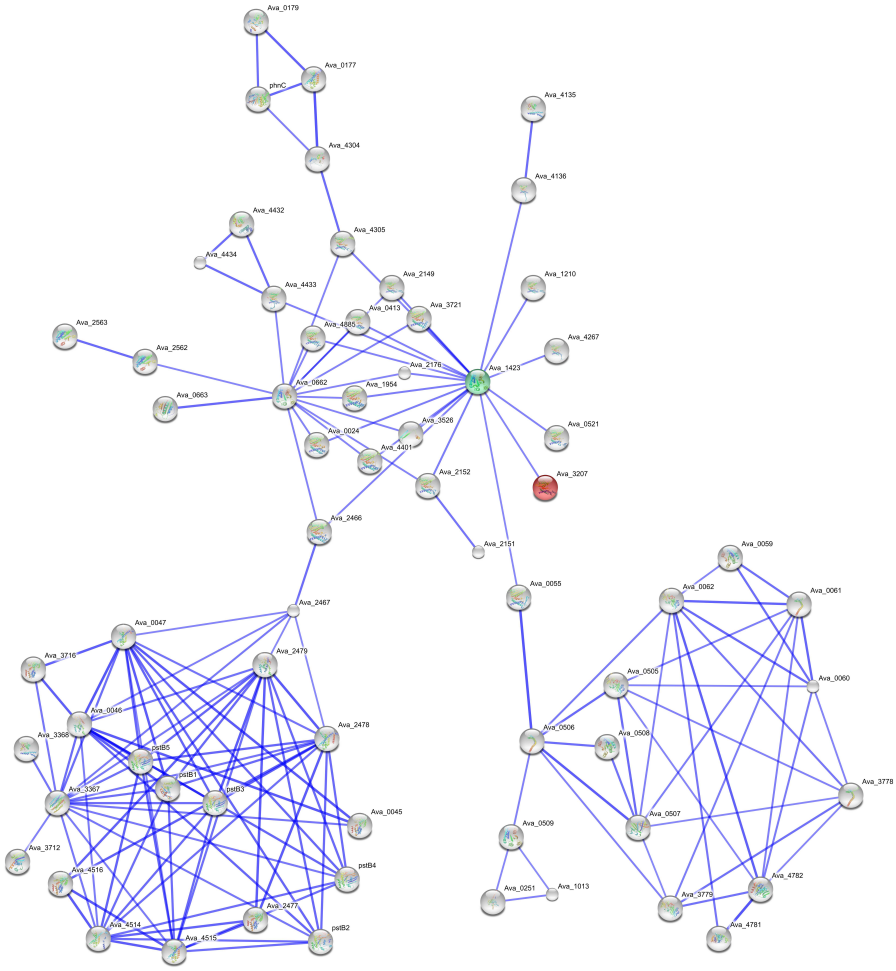


Figure 4.6: An example of the protein-protein interaction networks.

tropy based thermodynamic depth developed in Section 4.5 can characterize the structural complexity of the PPIs. The PPIs dataset consists of networks which describe the interaction relationships between histidine kinase and other proteins. Histidine kinase is a key protein in the development of signal transduction. If two proteins have direct (physical) or indirect (functional) association, they are connected by an edge. Examples of the PPIs are illustrated in Figure 4.6 and Figure 4.7. There are 219 PPIs in this dataset and they are collected from 5 different kinds of bacteria with the following evolution order (from older

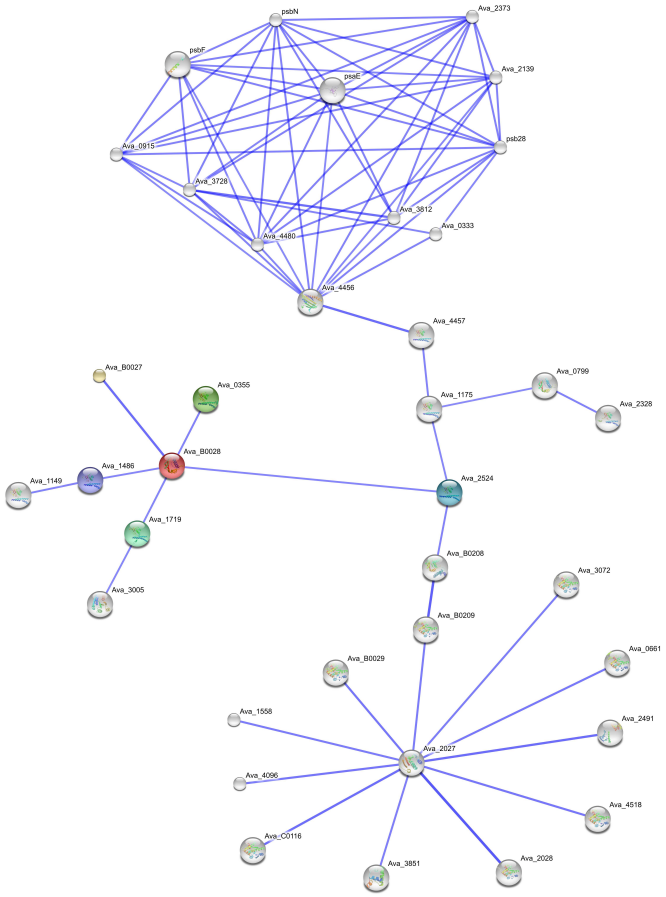


Figure 4.7: An example of the protein-protein interaction networks.

to more recent) *Aquifex* and *thermotoga*–8 PPIs from *Aquifex aelicus* and *Thermotoga maritima*, *Gram-Positive*–52 PPIs from *Staphylococcus aureus*, *Cyanobacteria*–73 PPIs from *Anabaena variabilis* and *Proteobacteria*–40 PPIs from *Acidovorax avenae*. There is an additional class (*Acidobacteria*–46 PPIs) which is more controversial in terms of the bacterial evolution since they were discovered. Although there are studies which relate many of them to different sub-phyla of the *Proteobacteria*, some of them have recently been placed very early in the phylogenic tree.

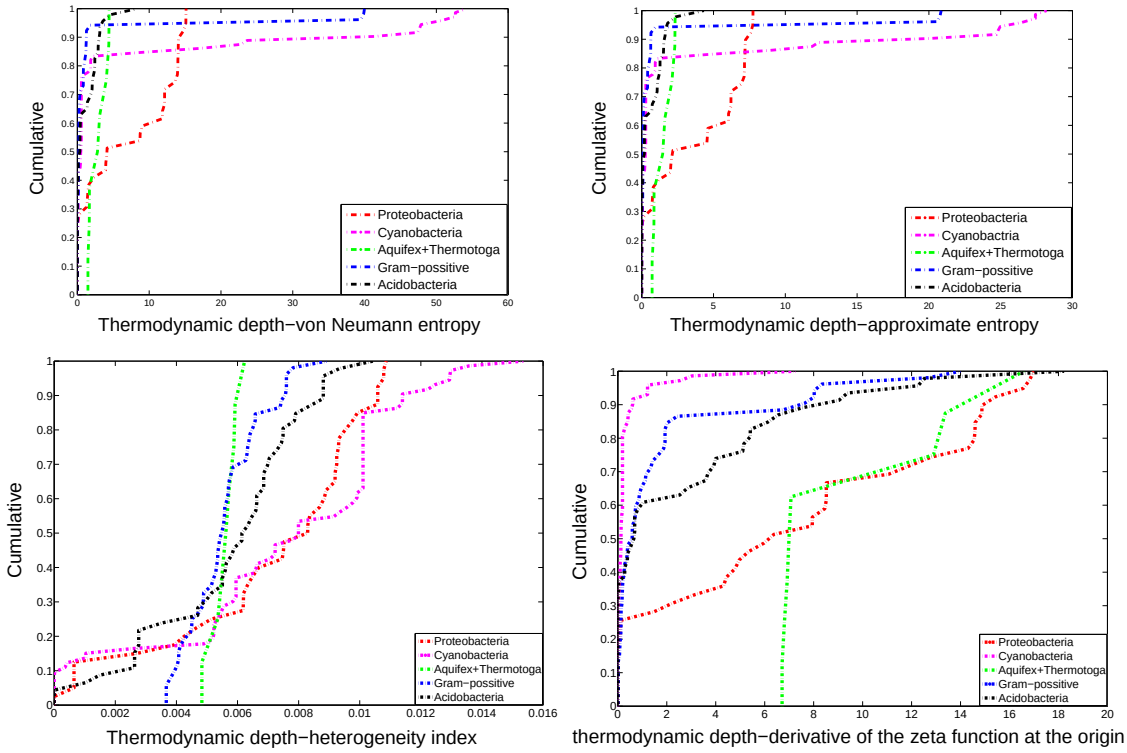


Figure 4.8: Cumulatives for: von Neumann entropy (top-left), approximate entropy (top-right), heterogeneity index (bottom-left) and derivative of the zeta function at the origin (bottom-right).

The question of whether the von Neumann entropy based thermodynamic depth is a good measure of the structural complexity of the PPIs can be answered by studying the cumulative distribution of the thermodynamic depth complexity [35]. From an evolutionary perspective, older (less evolved) bacteria have simpler PPIs and thus lower thermodynamic depth values compared with bacteria that have evolved more recently. This observation motivates the measurement of the area under the cumulative entropy distribution (CED). The greater the CED the simpler the PPIs. For purpose of comparison, we have also explored using the alternative three characterizations, i.e. the approximate entropy, the heterogeneity index and the derivative of the zeta function at the origin, as basic measures in the thermodynamic depth approach. The cumulative distributions of the

four thermodynamic depth measures are shown in Figure 4.8 and their corresponding area under the cumulatives are shown in Table 4.1 where the incorrect (inconsistent with evolution) values are shown in bold. The analysis of the area under the cumulatives in Table 4.1 gives the following results. The two entropy based thermodynamic depth measures overestimate the complexity of *Aquifex-Thermotoga*, whereas the heterogeneity index based thermodynamic depth overestimates the complexity of *Cyanobacteria*. The derivative of the zeta function at the origin overestimates the complexity of *Aquifex-Thermotoga* and underestimates that for *Cyanobacteria*. Finally, for the controversial *Acidobacteria*,

Table 4.1: Values of the area under the cumulatives of the four measures. According to the evolution order of bacteria which the PPIs are from, the order of the PPIs from simple to more complex are: *Aquifex-thermotoga*, *Gram-Positive*, *Cyanobacteria* and *Proteobacteria*, with a controversial class *Acidovorax avenae*. The simpler the PPIs, the greater the area under the cumulatives. The values that are not consistent with evolution order are shown in bold.

Bacteria	VNE[1]	AE[2]	Heterogeneity	zeta function derivative
Aquifex-Thermotoga	**95.40**%	**95.23**%	65.45%	**49.71**%
Gram-possitive	96.24%	96.09%	65.36%	90.65%
Cyanobacteria	89.27%	88.89%	**54.05**%	**98.31**%
Proteobacteria	88.82%	88.53%	56.45%	60.94%
Acidobacteria	98.22%	98.15%	65.84%	85.42%

[1] von Neumann entropy

[2] approximate entropy

the two entropy based thermodynamic depth measures and the heterogeneity index based thermodynamic depth place it oldest, whereas the derivative of the zeta function based measure places its order later than *Gram-possitive*. We note from those results that when

combined with the thermodynamic depth, the two entropy characterizations provide comparable results with the heterogeneity index based measure and also outperform the zeta function based measure.

4.7 Conclusions

In this chapter we have developed new graph characterizations from the von Neumann entropy. We commence from the von Neumann entropy of a graph. This is simply the entropy of density matrix associated with the normalized Laplacian matrix. We explore how to simplify and approximate the calculation of von Neumann entropy. Our first step is to replace the entropy by its quadratic counterpart. An analysis of the quadratic entropy reveals that it can be computed from a number of permutation invariant matrix trace expressions. This leads to a simple expression for the approximate entropy in terms of the elements of the degree matrix, and which can be computed without evaluating the normalized Laplacian matrix. Then we compare the new graph characterizations with their alternatives, i.e. Estrada's heterogeneity index and Riemann Zeta Function derivative, and we reveal a new link between Estradas index and the commute time on a graph. Finally, we introduce the entropy based thermodynamic depth as a graph complexity measure.

Experimental results on both synthetic dataset and real-world dataset reveal the approximate entropy is a good approximation of the von Neumann entropy for the BA, ER and Delaunay graphs. We have also compared the performance of six graph characterizations, i.e. the von Neumann entropy, the approximate entropy, the heterogeneity index and the derivative of the Riemann zeta function at the origin, the average path length and graph diameter, for distinguishing graphs. Here we observe that the two entropy methods give a better classification rate than the alternatives. In the final experiment, we investigated how to use the von Neumann entropy based thermodynamic depth to characterize the complexity of networks. This gives good results in ordering the PPIs of different

species of bacteria according to their evolved state.

Chapter 5

Generative Graph Prototypes from Information Theory

In this chapter, we combine the probabilistic framework introduced in Chapter 3 and the entropy-based graph characterization measures proposed in Chapter 4 and take an information theoretic method to construct a generative model for graphs by adopting a minimum description length approach. Here again the generative model is posed in the form of a prototype graph called supergraph. The complexity of the supergraph is encoded using the simplified von Neumann entropy (refer back to Equation (4.10) in Chapter 4). We develop a variant of the EM algorithm to minimize the description length. To generate new graphs, rather than only control the edge occurrence probabilities (as shown in the generative model developed in Chapter 3), we assume that both the nodes and the edges of graphs arise under independent Bernoulli distributions and sample new graphs according to their node and edge occurrence probabilities. Empirical evaluations on real-world database demonstrate the practical utility of the proposed algorithm and show the effectiveness of the generative model for the tasks of graph classification, graph clustering and generating new sample graphs.

5.1 Introduction

Given a set of sample graphs, we aim to learn a supergraph that best explains the graphs. The best supergraph model should be able to summarize the observed data well, and moreover, it should have good predictive capabilities. To locate the structure of this supergraph model, we take an information theoretic approach using a two-part minimum description length criterion [82] [79][80]. The two-part minimum description length (MDL) measures both the goodness-of-fit with the observed sample graphs under a supergraph model and the complexity of the supergraph. By trading off the first quantity against the second, it avoids overfitting the supergraph model. Torsello and Hancock [99] have shown how to learn a tree-union for a set of trees using the minimum description length criterion. Since the trees are rooted their learning process can be effected by performing tree merging operations in polynomial time. However, this greedy strategy does not translate tractably to graphs where the complexity becomes exponential, and we require different strategies for learning and sampling. Torsello and Hancock realize both objectives using edit operations. Here on the other hand we use a soft assignment method for optimization and then generate new instances using a direct sampling method.

To furnish the required learning framework, we adopt the probability distribution described in Chapter 3. This probability distribution is used to describe the likelihood of the sample graphs. To adopt the two-part minimum description length criterion, we also need a complexity measure of the supergraph. In traditional statistical models based on vector patterns, the complexity of the model is generally measured by counting the number of parameters in the model. However, this does not generalize well for graphs because information such as the number of edges or nodes of a graph is not sufficient to reflect its true complexity. Here we use an alternative measure of complexity, encoded using the von Neumann entropy proposed in Chapter 4 (i.e. the entropy of density matrix associated with its normalized Laplacian). We develop a variant of the EM algorithm to minimize the total code-length criterion. Here the structure of the supergraph and the correspondences

between the nodes of the sample graphs and those of the supergraph are treated as missing data. In the maximization step, we update both the node correspondence information and the structure of the supergraph using soft assignment [48]. After several iterations the variant EM algorithm will locate the structure of the supergraph that minimizes the overall-code length.

Besides developing a method of learning the structure of the supergraph model, we also investigate how to combine the Jensen-Shannon divergence with our supergraph to measure graph similarities. This investigation provides us a route to embed graph data into pattern space to perform graph clustering. Moreover, we also develop a novel and efficient method which allows our supergraph model to sample new graphs. This is realized by assuming the nodes and edges of sample graphs arise under Bernoulli distributions and we sample new graphs according to their node and edge occurrence probabilities. Therefore, our supergraph model proposed here can fulfil the tasks of graph classification, graph clustering and generating new graphs.

The remainder of this chapter is laid out as follows. In Section 5.2, we recall the probabilistic ingredients mentioned in Chapter 3, which describe the distribution of the graph data and are the prerequisites for our method. In Section 5.3, we explain how we encode our model so as to formulate the problem in hand in a minimum description length setting. In Section 5.4, we present a variant of the EM algorithm to minimize the code-length criterion. Section 5.5 exploits how to measure graph similarities using the Jensen-Shannon kernel and Section 5.6 shows how to sample new graphs from the generative model. Section 5.7 provides experiments to demonstrate the utility of our proposed algorithms. We first validate our variant EM algorithm by showing that the overall code-length decreases during the iterations. We then illustrate that our generative model outperforms alternative supergraph constructions on graph classification tasks. We also investigate the performance of graph clustering with the Jensen-Shannon kernel and explore to what extent the graphs sampled by our method reproduce the salient properties

of the original graphs used to train the supergraph model. Finally, Section 5.8 offers some conclusions.

5.2 Probabilistic Framework

To commence our development, we first recall the probabilistic framework we used to construct the supergraph in Chapter 3. We represent the set of sample graphs using $\mathcal{G} = \{G_1, \ldots, G_i, \ldots, G_N\}$, where the graph indexed i is $G_i = (V_i, E_i)$, with V_i as the node-set and E_i as the edge-set. Similarly, the supergraph which we aim to learn from this data is denoted by $\Gamma = (V_\Gamma, E_\Gamma)$, with node-set V_Γ and edge-set E_Γ. Furthermore, the structure of the sample graph G_i is represented using a $|V_i| \times |V_i|$ adjacency matrix $\mathbf{A}^i$ and the structure of the supergraph model Γ is represented using a $|V_\Gamma| \times |V_\Gamma|$ adjacency matrix $\mathbf{M}$. The elements of the adjacency matrix for the sample graph and those for the supergraph are respectively defined to be

$$A^i_{ab} = \begin{cases} 1 & \text{if } (a,b) \in E_i \\ 0 & \text{otherwise,} \end{cases} \qquad M_{\alpha\beta} = \begin{cases} 1 & \text{if } (\alpha,\beta) \in E_\Gamma \\ 0 & \text{otherwise.} \end{cases} \tag{5.1}$$

The correspondence information between the nodes of the sample graph and the nodes of the supergraph is represented using a $|V_i| \times |V_\Gamma|$ assignment matrix $\mathbf{S}^i$ which has elements

$$s^i_{a\alpha} = \begin{cases} 1 & \text{if } a \rightarrow \alpha \\ 0 & \text{otherwise,} \end{cases} \tag{5.2}$$

where $a \rightarrow \alpha$ implies that node $a \in V_i$ is matched to node $\alpha \in V_\Gamma$.

With the above ingredients, the *a posteriori* probability of the sample graph G_i given the structure of the supergraph and the node correspondences between each sample graph and the supergraph is [61]

$$P(G_i|\Gamma,\mathbf{S}^i)=\prod_{a\in V_i}\sum_{\alpha\in V_\Gamma}K_a^i\exp[\mu\sum_{b\in V_i}\sum_{\beta\in V_\Gamma}A_{ab}^iM_{\alpha\beta}s_{b\beta}^i], \tag{5.3}$$

where

$$\mu=\ln\tfrac{1-P_e}{P_e},\qquad K_a^i=P_e^{|V_i|\times|V_\Gamma|}B_a^i. \tag{5.4}$$

In the above, P_e is the error rate for node correspondence and B_a^i is the probability of observing node a in graph G_i, the value of which depends only on the identity of the node a, and $|V_i|$ and $|V_\Gamma|$ are the number of the nodes in graph G_i and supergraph Γ.

5.3 Model Coding Using MDL

With the probabilistic framework in hand, we take an information theoretic approach to estimating the structure of the supergraph Γ that best fits the set of sample graphs $\mathcal{G}$ by using a minimum description length criterion. Underpinning minimum description length is the principle that learning, or finding a model that explains some observed data and makes predictions about data yet unseen, can be viewed as finding a shortest code for the observed data [82] [79]. In its earliest realization introduced by Rissanen [80], the minimum description length principle states that the best model to explain a set of data is the one which minimizes the description length of the model together with the description length of the data, when encoded subject to the model. To formalize this idea, we encode and transmit the data together along with the model. In our case these are respectively the sample graphs $\mathcal{G}$ and the supergraph structure Γ. This leads to a two-part message whose total length is given by

$$\mathcal{L}(\mathcal{G},\Gamma)=LL(\mathcal{G}|\Gamma)+LL(\Gamma). \tag{5.5}$$

where $LL(\mathcal{G}|\Gamma)$ is the code-length of the sample graphs given the supergraph and $LL(\Gamma)$ is the code-length of the supergraph. Then the optimal supergraph is the one that minimizes this total code-length. By taking into account the total code-length in the model, MDL allows us to select a supergraph representation that trades-off goodness-of-fit with the observed sample graphs against the complexity of the model.

5.3.1 Encoding Sample Graphs

To apply the two-part MDL principle, we first compute the code-length of the graph data. A general choice for the code-length of the graph data is the *Shannon-Fano code* [30] which is equivalent to the negative logarithm of its likelihood function given the supergraph. Instead of using the *Shannon-Fano code*, here we measure the code-length of the graph data using its average. Our reason is that if we adopt the former measure, then there is a bias to learning a complete supergraph that is fully connected. The reason will become clear later-on when we outline the maximization algorithm in Section 5.4, and we defer our justification until later. To compute the likelihood of the graph data, for the sample graph-set $\mathcal{G} = \{G_1, \ldots G_i, \ldots G_N\}$ and the supergraph Γ, we use $\mathcal{S} = \{\mathbf{S}^1, \ldots \mathbf{S}^i, \ldots \mathbf{S}^N\}$ to represent the set of assignment matrices and these indicate the correspondences between the nodes of the sample graphs and those of the supergraph. Under the assumption that the graphs in $\mathcal{G}$ are independent samples from the distribution, using the *a posteriori* probability from Section 5.2 the likelihood of the set of sample graphs is

$$P(\mathcal{G}|\Gamma,\mathcal{S}) = \prod_{G_i\in\mathcal{G}} P(G_i|\Gamma,\mathbf{S}^i) = \prod_{G_i\in\mathcal{G}}\prod_{a\in V_i}\sum_{\alpha\in V_\Gamma} K_a^i \exp[\mu \sum_{b\in V_i}\sum_{\beta\in V_\Gamma} A_{ab}^i M_{\alpha\beta} s_{b\beta}^i]. \quad (5.6)$$

Then the graph code-length is

$$LL(\mathcal{G}|\Gamma) = -\frac{1}{|\mathcal{G}|}\ln P(\mathcal{G}|\Gamma,\mathcal{S}) = -\frac{1}{|\mathcal{G}|}\sum_{G_i\in\mathcal{G}}\sum_{a\in V_i}\ln\{\sum_{\alpha\in V_\Gamma}K_a^i\exp[\mu\sum_{b\in V_i}\sum_{\beta\in V_\Gamma}A_{ab}^i M_{ab}s_{b\beta}^i]\}, \tag{5.7}$$

which is the average over the set of sample graphs $\mathcal{G}$.

5.3.2 Encoding the Supergraph Model

Next, we need to compute a code-length to measure the complexity of the supergraph. For two-part codes the MDL principle does not give any guideline as to how to encode the hypotheses. Hence every code for encoding the supergraph structure is allowed, so long as it does not change with the sample size N. Graph characterizations such as the number of edges or nodes can express some properties of graphs, however they are not sufficient to reflect the true complexity of the graphs. Thus we need to seek for a more meaningful measure of graph complexity. Here we use the von Neumann entropy associated with the normalized Laplacian matrix we proposed in Chapter 4 to give a code-length for the supergraph complexity. According to Equation (4.4) in Chapter 4, the von Neumann entropy of the supergraph Γ is defined as

$$H(\Gamma) = -\sum_{j=1}^{|V_\Gamma|}\frac{\hat{\lambda}_j}{|V_\Gamma|}\ln\frac{\hat{\lambda}_j}{|V_\Gamma|},$$

where $|V_\Gamma|$ is the number of nodes in the supergraph and $\hat{\lambda}_j$ are the eigenvalues of the normalized Laplacian matrix of the supergraph. To incorporate the supergraph complexity with the code-length of the graph data, we need to express the von Neumann entropy in terms of the simple statistics for the graph, as in the code-length expression. Fortunately, we have shown in Chapter 4 that replacing the Shannon entropy by the quadratic entropy and using some transformations, the von Neumann entropy can be approximated in terms of node degree statistics

$$\bar{H}(\Gamma) = 1 - \frac{1}{|V_\Gamma|} - \sum_{(\alpha,\beta)\in E_\Gamma} \frac{1}{|V_\Gamma|^2\, d_\alpha d_\beta}\,, \tag{5.8}$$

where E_Γ is the edge-set of the supergraph and d_α and d_β are the degree of nodes α and β of the supergraph. Finally, by adding together the two contributions to the code-length, the overall code-length is

$$\mathcal{L}(\mathcal{G},\Gamma) = LL(\mathcal{G}|\Gamma) + LL(\Gamma) = \tag{5.9}$$
$$-\frac{1}{|\mathcal{G}|}\sum_{G_i\in\mathcal{G}}\sum_{a\in V_i}\ln\{\sum_{\alpha\in V_\Gamma} K_a^i \exp[\mu\sum_{b\in V_i}\sum_{\beta\in V_\Gamma} A_{ab}^i M_{ab} s_{b\beta}^i]\,\} + 1 - \frac{1}{|V_\Gamma|} - \sum_{(\alpha,\beta)\in E_\Gamma}\frac{1}{|V_\Gamma|^2\, d_\alpha d_\beta}\,.$$

Unfortunately, due to the mixture structure, the direct estimation of the supergraph structure $\mathbf{M}$ from the above code-length criterion is not tractable in closed-form. For this reason, we resort to using the EM algorithm.

5.4 The Expectation-Maximization Algorithm

Having developed our computational model which poses the problem of learning the supergraph as that of minimizing the code-length, in this section, we provide a concrete algorithm to locate the supergraph structure using our code-length criterion. The minimization of the code-length is equivalent to the maximization of its negative, and we develop an EM algorithm to realize the maximization. We view the node correspondence information between the sample graphs and supergraph as missing data, and regard the structure of the supergraph as the set of parameters to be estimated. The initialization of the EM requires an initial supergraph structure and an initial correspondence between the sample graphs and the initial supergraph. In the two interleaved steps of the EM algorithm, the expectation step involves recomputing the *a posteriori* probability of node correspondence while the maximization step involves updating both the structure of the

supergraph and the node correspondence information. After each maximization step, we recompute the value of the code-length using the updated information of the supergraph structure and the node correspondences. When the difference between the new value of the code-length and the old value of the code-length are always smaller than a set threshold value (normally a very small positive value), it means the code-length converges. Otherwise, we continue interleaving the two steps of the EM algorithms. In the experimental part, we will initialize the supergraph using different structures and investigate their convergence.

5.4.1 Weighted Code-length Function

To compute the weighted log-likelihood of the overall code-length, we make use of Luo and Hancock's log-likelihood function for correspondence matching. According to Luo and Hancock [61], treating the assignment matrix as missing data, the weighted log-likelihood function for observing a sample graph G_i, i.e. for it to have been generated by the supergraph Γ is

$$\Lambda^{(n+1)}(G_i|\Gamma, \mathbf{S}^{i,(n+1)}) = \sum_{a\in V_i}\sum_{\alpha\in V_\Gamma} Q_{a\alpha}^{i,(n)}\{\ln\ K_a^i + \mu\sum_{b\in V_i}\sum_{\beta\in V_\Gamma} A_{ab}^i M_{\alpha\beta}^{(n)} s_{b\beta}^{i,(n+1)}\}, \quad (5.10)$$

where the superscript n indicates that the quantity is taken at iteration n of the EM algorithm and $\mathbf{Q}^{i,(n)}$ is a matrix with elements $Q_{a\alpha}^{i,(n)}$ that are set equal to the *a posteriori* probability of node *a* in G_i being matched to node α in Γ at iteration n of the EM algorithm.

With the above likelihood function and the code-length developed in the previous

section, the EM algorithm involves maximizing

$$\bar{\Lambda}^{(n+1)}(\mathcal{G}|\Gamma,\mathcal{S}^{(n+1)}) = \frac{1}{|\mathcal{G}|}\sum_{G_i\in\mathcal{G}}\sum_{a\in V_i}\sum_{\alpha\in V_\Gamma} Q_{a\alpha}^{i,(n)}\{\ln K_a^i + \mu\sum_{b\in V_i}\sum_{\beta\in V_\Gamma} A_{ab}^i M_{\alpha\beta}^{(n)} s_{b\beta}^{i,(n+1)}\}$$
$$-1+\frac{1}{|V_\Gamma|}+\sum_{(\alpha,\beta)\in E_\Gamma}\frac{1}{|V_\Gamma|^2\, d_\alpha d_\beta}\,. \tag{5.11}$$

The expression above can be simplified since the first term under the curly braces contributes a constant amount

$$\sum_{G_i\in\mathcal{G}}\sum_{a\in V_i}\sum_{\alpha\in V_\Gamma} Q_{a\alpha}^{i,(n)}\ln K_a^i = \sum_{G_i\in\mathcal{G}}\sum_{a\in V_i}\ln K_a^i. \tag{5.12}$$

Based on this observation, the critical quantity in determining the update direction is

$$\hat{\Lambda}^{(n+1)} = \frac{1}{|\mathcal{G}|}\sum_{G_i\in\mathcal{G}}\sum_{a\in V_i}\sum_{\alpha\in V_\Gamma}\sum_{b\in V_i}\sum_{\beta\in V_\Gamma} Q_{a\alpha}^{i,(n)} A_{ab}^i M_{\alpha\beta}^{(n)} s_{b\beta}^{i,(n+1)} - 1 + \frac{1}{|V_\Gamma|} + \sum_{(\alpha,\beta)\in E_\Gamma}\frac{1}{|V_\Gamma|^2\, d_\alpha d_\beta}. \tag{5.13}$$

5.4.2 Maximization

In order to optimize our weighted code-length criterion, we use graduated assignment [48] to update both the assignment matrices $\mathcal{S}$ and the structure of the supergraph, i.e. the supergraph adjacency matrix $\mathbf{M}$. The updating process is realized by computing the derivatives of $\hat{\Lambda}^{(n+1)}$, and reformulating the underlying discrete assignment problem as a continuous one using soft assignment [18].

In the maximization step, we have two parallel iterative update equations. The first update mode involves softening the assignment variables, while the second aims to modify the edge structure in the supergraph. Supergraph edges that are unmatchable become disjoint by virtue of having weak connection weights and cease to play any significant role in the update process. Experiments show that the algorithm appears to be numerically stable and appears to converge uniformly.

Updating Assignment Matrices: To update the assignment matrices, we commence by computing the partial derivative of the weighted code-length function in Equation (5.13) with respect to the elements of the assignment matrices, which gives

$$\frac{\partial \hat{\Lambda}^{(n+1)}}{\partial s_{b\beta}^{i,(n+1)}} = \frac{1}{|\mathcal{G}|} \sum_{a \in V_i} \sum_{\alpha \in V_\Gamma} Q_{a\alpha}^{i,(n)} A_{ab}^{i} M_{\alpha\beta}^{(n)}. \tag{5.14}$$

To ensure that the assignment variables remain constrained to lie within the range [0,1], we adopt the soft assingment update rule

$$s_{a\alpha}^{i,(n+1)} \longleftarrow \frac{\exp[\varepsilon \frac{\partial \hat{\Lambda}^{(n+1)}}{\partial s_{a\alpha}^{i,(n+1)}}]}{\sum_{\alpha' \in V_\Gamma} \exp[\varepsilon \frac{\partial \hat{\Lambda}^{(n+1)}}{\partial s_{a\alpha'}^{i,(n+1)}}]}. \tag{5.15}$$

The value of ε in the update process has been controlled using a slow exponential annealing schedule of the form suggested by Gold and Rangarajan [48]. Initializing ε with a small positive value and allowing it to gradually increase, the assignment variable $s_{a\alpha}^{i,(n+1)}$ corresponding to the maximum $\frac{\partial \hat{\Lambda}^{(n+1)}}{\partial s_{a\alpha}^{i,(n+1)}}$ approaches 1 while the remainder approach 0.

Updating Supergraph Structure: The partial derivative of the weighted code-length function in Equation (5.13) with respect to the elements of the supergraph adjacency matrix is equal to

$$\frac{\partial \hat{\Lambda}^{(n+1)}}{\partial M_{\alpha\beta}^{(n)}} = \frac{1}{|\mathcal{G}|} \sum_{G_i \in \mathcal{G}} \sum_{a \in V_i} \sum_{b \in V_i} Q_{a\alpha}^{i,(n)} A_{ab}^{i} s_{b\beta}^{i,(n+1)} - \frac{1}{|V_\Gamma|^2 (d_\alpha^{(n)})^2} \sum_{(\alpha,\beta') \in E_\Gamma} \frac{1}{d_{\beta'}^{(n)}}. \tag{5.16}$$

The soft assignment update equation for the elements of the supergraph adjacency matrix is

$$M_{\alpha\beta}^{(n+1)} \longleftarrow \frac{\exp[\varepsilon \frac{\partial \hat{\Lambda}^{(n+1)}}{\partial M_{\alpha\beta}^{(n)}}]}{\sum_{(\alpha',\beta') \in E_\Gamma} \exp[\varepsilon \frac{\partial \hat{\Lambda}^{(n+1)}}{\partial M_{\alpha'\beta'}^{(n)}}]}. \tag{5.17}$$

In the case of the updating of the assignment matrix elements, in each row and each column of the recovered assignment matrix no more than one element can take on unit value. By contrast, in the case of the recovered supergraph adjacency matrix there may exist multiple elements in each row or column with a unit value. To deal with this problem, in practice we set a threshold, and then recover the adjacency matrix by setting all elements larger than the threshold to unity and by setting the remaining elements to zero. This is repeated each time we increase the value of ε in the annealing schedule.

From Equation (5.16), it is interesting to note that the derivatives of $\hat{\Lambda}^{(n+1)}$ with respect to the elements of the supergraph adjacency matrix are dependent on the frequency of sample-set edges that are in correspondence with the same supergraph edge. To illustrate this point, if we approximate the matrix $\mathbf{Q}$ using $\mathbf{S}$, then the first term in Equation (5.16) becomes the expectation value of the permutated adjacency matrices for the sample graphs. As a result, the elements of the supergraph adjacency matrix reflect the frequency of corresponding edges in the sample-set. The thresholding process selects frequent edges and removes infrequent ones.

Recall that in Section 5.3.1 we discussed the encoding of the sample graphs, and chose to use the average of the *Shannon-Fano code-length*. We can now elucidate that the reason for this choice is that as the number of the sample graphs increases, for instance in the limit as the size of the graph sample-set $\mathcal{G}$ increases, i.e. $N \rightarrow \infty$, the sum of permuted adjacency matrices of the sample graphs might dominate the magnitude of the second term in Equation (5.16). Thus the update algorithm might induce a complete supergraph that is fully connected. Hence, we choose to use its average rather than its sum.

5.4.3 Expectation

In the expectation step of the EM algorithm, we compute the *a posteriori* probabilities of the nodes in the supergraph being matched to the nodes in the sample graphs. Applying Bayes rule, the *a posteriori* probabilities of the nodes in the supergraph corresponding to

the nodes in the sample graph G_i at iteration $n+1$ are given by

$$Q_{a\alpha}^{i,(n+1)} = \frac{\exp[\sum_{b\in V_i}\sum_{\beta\in V_\Gamma} A_{ab}^i M_{\alpha\beta}^{(n)} s_{b\beta}^{i,(n)}]\pi_\alpha^{i,(n)}}{\sum_{\alpha'\in V_\Gamma}\exp[\sum_{b\in V_i}\sum_{\beta\in V_\Gamma} A_{ab}^i M_{\alpha\beta}^{(n)} s_{b\beta}^{i,(n)}]\pi_{\alpha'}^{i,(n)}} , \tag{5.18}$$

where

$$\pi_{\alpha'}^{i,(n)} = \frac{1}{|V_i|}\sum_{a\in V_i} Q_{a\alpha'}^{i,(n)}. \tag{5.19}$$

5.5 Information Theoretic Kernel

The information theoretic formulation presented in this chapter also provides a natural route to the kernelized analysis of graph similarity, since the measure of the von Neumann entropy can be used to construct an information theoretic kernel. The route we take here is to form supergraphs from pairs of graphs, and then to compute the so-called Jensen-Shannon (JS) divergence [23] [59] between graphs as an information theoretic measure of dissimilarity. The JS divergence is found by taking the difference between the entropy of the pairwise supergraph and the average of the separate entropies of the two graphs used to construct it. The JS divergence is used to construct the information theoretic and non-extensive Jensen-Shannon kernel [66]. More specifically, we measure the dissimilarity between graphs using the JS divergence

$$JS(G_i, G_j) = H(G_i \oplus G_j) - \frac{H(G_i) + H(G_j)}{2} . \tag{5.20}$$

In the above equation, $G_i \oplus G_j$ represents the union for graphs G_i and G_j, and $H(\cdot)$ denotes the entropy of the corresponding graph. From the Jensen-Shannon divergence we construct a kernel $K(G_i, G_j) = \ln 2 - JS(G_i, G_j)$ and with the kernel matrix to hand we embed the graphs into pattern space using kernel principal component analysis (kernel PCA).

The supergraph learning method proposed in this chapter exploits a method for computing the Jensen-Shannon divergence between pairs of graphs. To do this, we use our supergraph learning method to construct a graph-union $G_i \oplus G_j$ for every pair of graphs G_i and G_j. The graph-union $G_i \oplus G_j$ is the supergraph of G_i and G_j learned by using the minimum description length criterion. Using the von Neumann entropy as the entropy of graphs, we measure the similarities of the graphs using the Jensen-Shannon divergence and then embed graphs into pattern space using kernel PCA.

5.6 Sampling From the Generative Model

In this section we explore whether our generative model can be used to sample new graphs. Given the *a posterior* probability in Equation (5.3), the task of sampling graphs from the generative model is only tractable using a Monte Carlo technique. However, Monte Carlo sampling is computationally expensive since procedures such as edge insertion or deletion on a sample graph may affect the assingment matrix and may therefore take excessive amount of time to cycle through all the edges of the supergraph. Here we provide a direct sampling method, based on the assumption that graphs are drawn from a simple distribution. We assume that the nodes and edges of the sample graphs arise as independent samples from the supergraph under a Bernoulli distribution. Given the learned structure of the supergraph model $\hat{\Gamma}$ and the assignment matrices $\hat{\mathcal{S}}$ obtained from our EM algorithm, then the likelihood of the sampled graphs $\mathcal{G}$ becomes

$$P(\mathcal{G}|\hat{\Gamma},\hat{\mathcal{S}}) = \prod_{G_i\in\mathcal{G}} \prod_{\alpha,\beta\in V_{\hat{\Gamma}}} P_\alpha^{V\ \sum_{a\in V_i}\hat{s}^i_{a\alpha}}(1-P_\alpha^V)^{1-\sum_{a\in V_i}\hat{s}^i_{a\alpha}}\ P_{\alpha\beta}^{E\ \sum_{a,b\in V_i}\hat{s}^i_{a\alpha}\hat{s}^i_{b\beta}A^i_{ab}}(1-P_{\alpha\beta}^E)^{1-\sum_{a,b\in V_i}\hat{s}^i_{a\alpha}\hat{s}^i_{b\beta}A^i_{ab}} \tag{5.21}$$

where P_α^V is the probability that node α of the generative model Γ is present in the set of graphs $\mathcal{G}$ and $P_{\alpha\beta}^E$ is the conditional probability that edge (α,β) occurs when nodes α and

β are present in Γ. The trial success probability for the Bernoulli distributions P_α^V and $P_{\alpha\beta}^E$ is equal to the expected number of successes, and so

$$P_\alpha^V = \frac{1}{|\mathcal{G}|} \sum_{G_i \in \mathcal{G}} \sum_{a \in V_i} \hat{s}_{a\alpha}^i \,, \tag{5.22}$$

$$P_{\alpha\beta}^E = \frac{1}{|\mathcal{G}|} \sum_{G_i \in \mathcal{G}} \sum_{a,b \in V_i} \hat{s}_{a\alpha}^i \hat{s}_{b\beta}^i A_{ab}^i \,. \tag{5.23}$$

To generate a new graph from the distribution, we first sample nodes from the generative model using the node occurrence probabilities computed in Equation (5.22). To do this, for each node $\alpha \in V_{\hat{\Gamma}}$, we use a random generator to return a scalar value drawn from a uniform distribution on the interval [0,1] and compare the occurrence probability of the node P_α^V and the scalar value. If the occurrence probability of the node is larger than the scalar value, the node is selected to be present in the sample graph; otherwise, the node is not present. After we have sampled the nodes that are present in the generated graph, we decide whether there are edges between pairs of these present nodes. It is realized in a similar manner of the node sampling. That is, for each pair of the present nodes (α, β), we generate a random value drawn from the uniform interval [0,1] and compare their edge occurrence probability $P_{\alpha\beta}^E$ computed from Equation (5.23) with the random value. If their edge occurrence probability is greater than the random value, there will be an edge between this pair of nodes; otherwise, there will be no connection between them. Algorithm 1 gives the pseudo code for the sampling procedure.

Algorithm 1: Sampling Graphs From The Probabilistic Generative Model

Input: A generative model $\hat{\Gamma} = (V_{\hat{\Gamma}}, E_{\hat{\Gamma}})$ with probabilities P_{α}^{V} on each node $\alpha \in V_{\hat{\Gamma}}$ and $P_{\alpha\beta}^{E}$ on each corresponding edge $(\alpha, \beta) \in E_{\hat{\Gamma}}$

Output: Some sample graphs

1: Initialize a null sample graph $G_{SG} = (V_{SG}, E_{SG})$
2: For each node $\alpha \in V_{\hat{\Gamma}}$
3: If $P_{\alpha}^{V} > rand$
4: Add node α to V_{SG}
5: End
6: End
7: For each pair of nodes $(\alpha, \beta) \in V_{SG}$
8: If $P_{\alpha\beta}^{E} > rand$
9: Add edge (α, β) to E_{SG}
10: End
11: End
12: Delete the disconnected nodes in V_{SE}.
13: Repeat the above procedures until obtain some sample graphs.

Note: the $rand$ command generates a random value between 0 and 1 from a uniform distribution.

5.7 Experiments

In this section, we report experimental results aimed at demonstrating the utility of our proposed generative model on real-world data. We use images from two datasets for experiments. The first dataset is the COIL [68] which consists of images of four objects, with 72 views of each object from equally spaced directions over 360°. We extract corner features using the corner detector [54] from each image and use the detected feature points as nodes to construct sample graphs by Delaunay triangulation. The second "toys" dataset consists of views of toys, and contains images of 4 objects with 20 different views of each object. For this second dataset, the feature points used to construct Delaunay graphs are extracted using the SIFT [60] detector. Some example images of the objects and their associated Delaunay graphs from these two datasets are given in Figure 5.1. The experimental study with these datasets is divided into four parts. We commence by exploring the convergence properties of our supergraph learning algorithm, then we evaluate the performance of the our learned model on graph classification and graph clustering tasks. Finally we explore to what extent the sample graphs from the generative model reproduce the statistical properties of the original graphs used to train the supergraph model.

5.7.1 Convergence

The first part of our experimental investigation aims to explore the convergence properties of our supergraph learning method. We test our proposed algorithm on the COIL and "toys" datasets. We initialize the supergraph structure with the set median graph [54], i.e. the sample graph with the largest average of the *a posteriori* probabilities to the other sample graphs. Then we match the sample graphs from the two datasets against their respective initial supergraphs using graduated assignment [48] and initialize the assignment matrices in our algorithm with the resulting assignment matrices. Using these settings, we iterate the two steps of the EM algorithm, and observe how the complexity of the

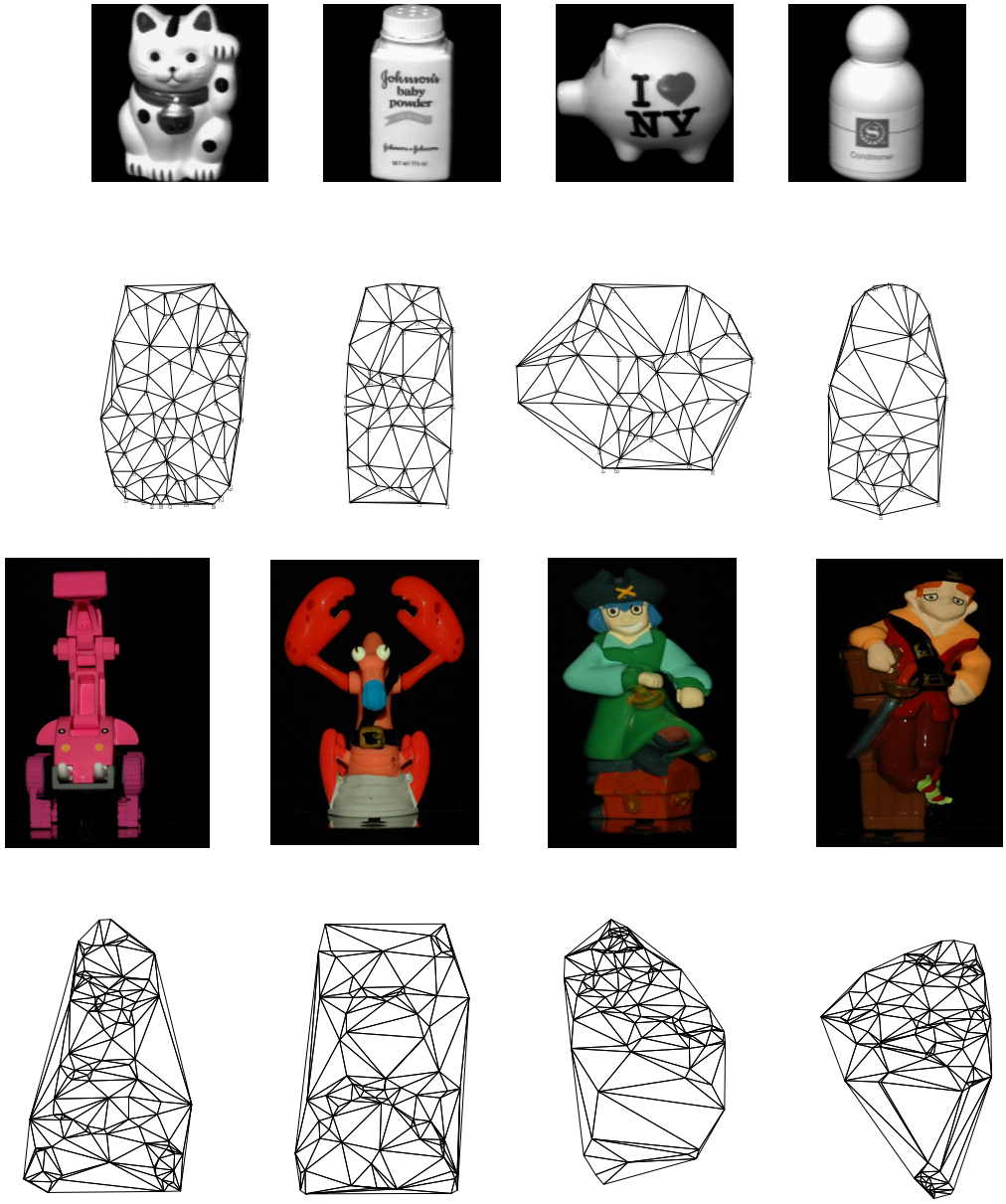


Figure 5.1: Example images and their associated graphs. Up two rows: COIL images and their associated graphs. Down two rows: Toy images and their associated graphs.

supergraph, the average log-likelihood of the sample graphs and the overall code-length vary with iteration number. Figures 5.2 and Figure 5.3 respectively show the results for the COIL and "toys" datasets illustrated in Figure 5.1.

Figure 5.2(a) and Figure 5.3(a) show the variations of the simplified von Neumann entropy for the two datasets, and from the figures it is clear that the simplified von Neumann entropy of the supergraph increases as the iteration number increases. This indicates that the supergraph structure becomes more complex with an increasing number of iterations. Figure 5.2(b) and Figure 5.3(b) show that the average of the log-likelihood of the sample graphs increases with the iteration number, while Figure 5.2(c) and Figure 5.3(c) show that the overall-code length decreases and gradually converges as the number of iterations increases.

In order to better analyze our method, we have also experimented with initializing the supergraph with different structures. This is effected using SIFT feature descriptors for the "toy" dataset. That is, we match pairs of the neighbour graphs using the SIFT feature descriptors and concatenate the common structures over the sample graphs from the same object to form an initial supergraph. The initial supergraph constructed in this way preserves more of the structural variations present in the set of sample graphs. Figure 5.4 shows the results obtained when we initialize using this concatenated supergraph. The figure shows how the three quantities studied in Figure 5.2 and Figure 5.3 change during the EM algorithm. Compared with the plots in Figure 5.2(a) and Figure 5.3(a), the von Neumann entropy in Figure 5.4(a) shows an opposite trend and decreases as the number of iterations increases. The reason for this is that the initial supergraph, i.e. the concatenated supergraph, accommodates too much structural variation from the sample graphs. The reduction of the simplified von Neumann entropy implies some trivial edges are eliminated or relocated. As a result the supergraph structure both condenses and simplifies with increasing iteration number. Although the complexity of the supergraph behaves differently, the average of the likelihood of the graphs in Figure 5.4(b) exhibits

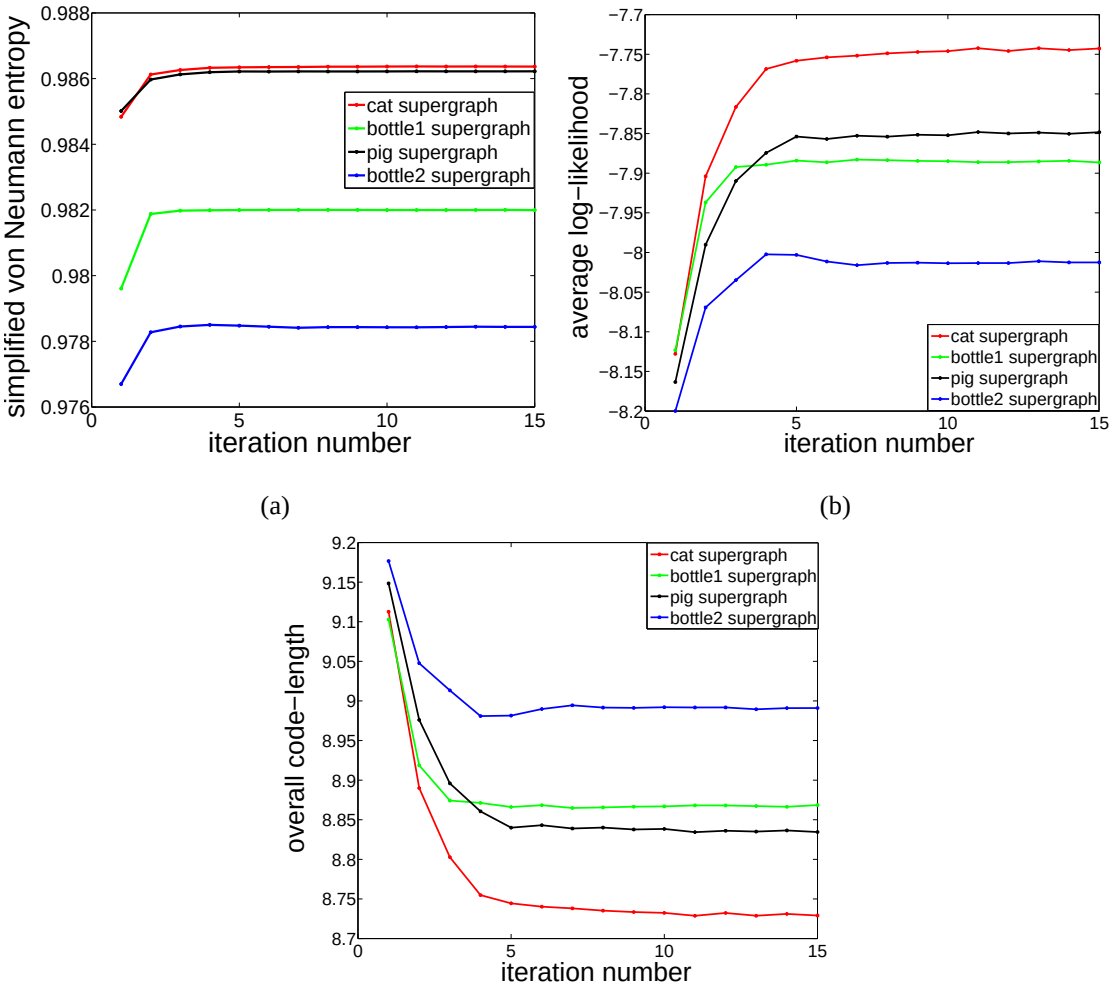


Figure 5.2: COIL dataset: (a) variation of the complexity of the supergraph, encoded as the simplified von Neumann entropy, during iterations, (b) variation of average log-likelihood of the sample graphs during iterations and (c) variation of the overall code-length during iterations.

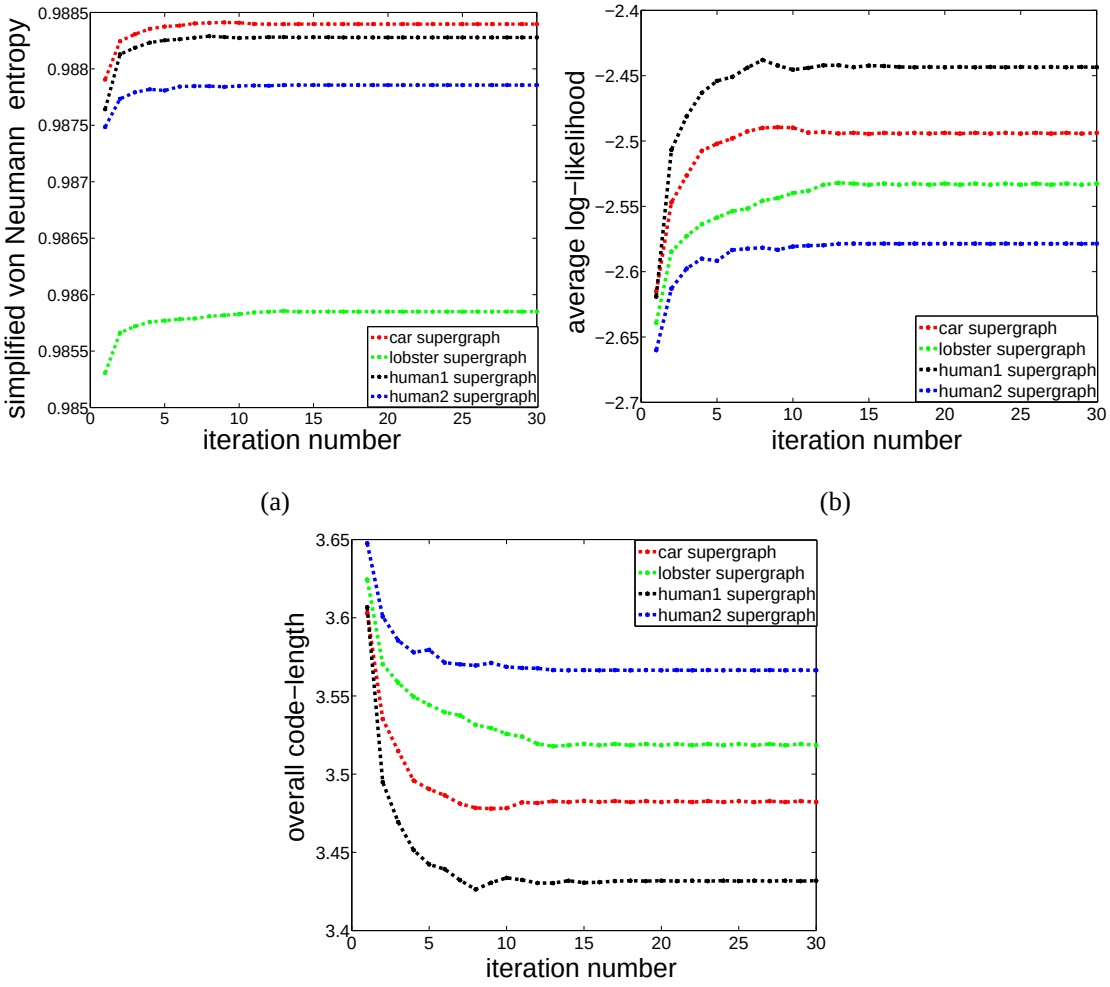
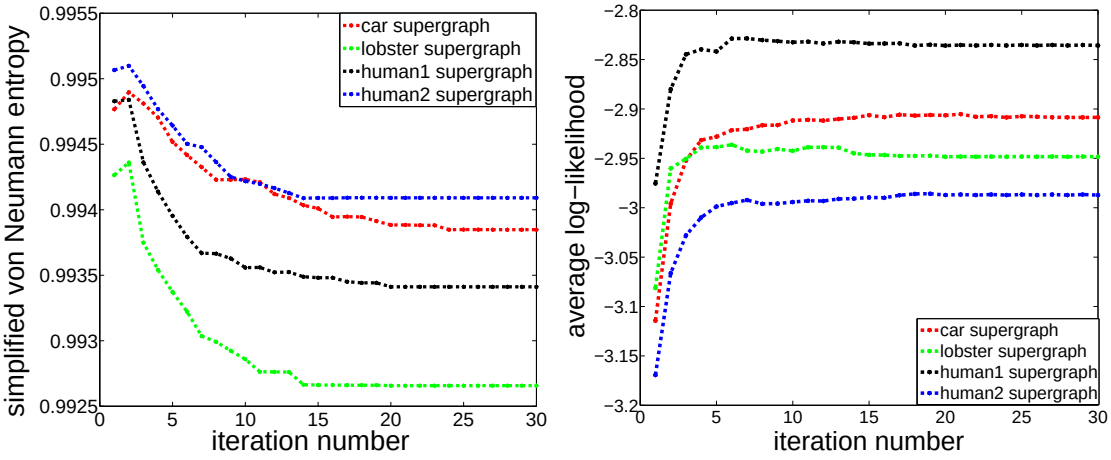
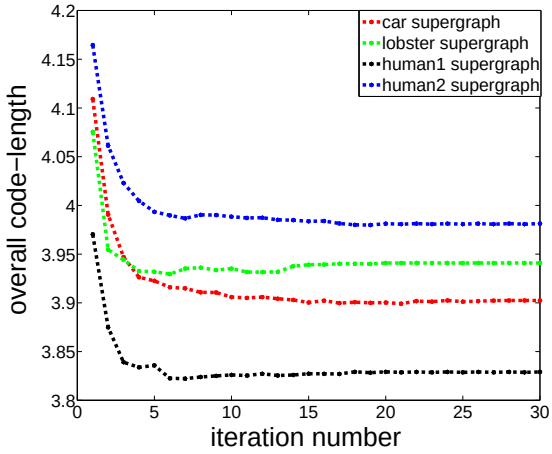


Figure 5.3: “Toys” dataset (The set median graph is used to initialize the EM algorithm): (a) variation of the complexity of the supergraph, encoded as the simplified von Neumann entropy, during iterations, (b) variation of the average log-likelihood of the sample graphs during iterations and (c) variation of the overall code-length during iterations.



(a) (b)



(c)

Figure 5.4: "Toys" dataset (The concatenated supergraph is used to initialize the EM algorithm): (a) variation of the complexity of the supergraph, encoded as the simplified von Neumann entropy, during iterations, (b) variation of the average log-likelihood of the sample graphs during iterations and (c) variation of the overall code-length during iterations.

a similar behaviour to those in Figure 5.2(b) and Figure 5.3(b) , and the overall-code length in Figure 5.4(c) has a similar behaviour to those in Figure 5.2(c) and Figure 5.3(c). In other words, our algorithm behaves in a stable manner both increasing the likelihood of sample graphs and decreasing the overall code-length on both dataset. We note that the structures of the supergraphs we learned in Figure 5.3 and Figure 5.4 are different. This is because the EM algorithm is sensitive to initializations. Since we initialize the supergraph using the set median graph in Figure 5.3 and the concatenated graph in Figure 5.4, the supergraphs we learned have different structures.

5.7.2 Classification

Our second experimental goal is to evaluate the effectiveness of our learned generative model for classifying out-of-sample graphs. From the COIL dataset, we aim 1) to distinguish images of cats from pigs on the basis of their graph representations and 2) distinguish between images of different types of bottles. For the "toys" dataset, on the other hand, we aim to distinguish between images of the four objects. To perform these classification tasks, we learn a supergraph for each object class from a set of samples and use Equation (5.3) to compute the *a posteriori* probabilities for each graph from a separate (out-of-sample) test-set. The class-label of the test graph is determined by the class of the supergraph which gives the maximum *a posteriori* probability. The classification rate is the fraction of correctly identified objects computed using 10-fold cross validation. To perform the 10-fold cross validation for the COIL dataset, we index the 72 graphs from a same object according to their image view direction from 0° to 360°, and in each instance we select 7 or 8 graphs that are equally spaced over the angular interval as test-set, and the remainder are used as as sample-set for training. We use a similar procedure for the "toys" dataset. For comparison, we have also investigated the results obtained using two alternative constructions of the supergraph. The first of these is the set median graph used to initialize our algorithm. The second is the supergraph learned without taking its com-

plexity into account. This supergraph is learned by maximizing the likelihood function of the sample graphs given in Equation (5.6). Table 5.1 shows the classification results obtained with our supergraph construction using minimum description length and the other two alternative supergraph constructions. From the three constructions, it is the supergraphs learned using the MDL principle that achieve the highest classification rates on all three classification tasks.

Table 5.1: Comparison of the classification results. We show the average classification rates from 10-fold cross validation and their standard error. The highest classification rates are shown in bold.

Classification Rate	cat & pig	bottle1 & bottle2	four objects (Toys)
learned supergraph (by MDL)	**83.2% ± 0.041**	**76.6% ± 0.027**	**75.2% ± 0.025**
learned supergraph[1]	80.7% ± 0.056	69.9% ± 0.029	72.5% ± 0.022
set median graph[2]	66.9% ± 0.052	65.1% ± 0.023	65.5% ± 0.025

[1] the supergraph learned using method from Chapter 3

[2] refer to [56]

5.7.3 Clustering

In this section we provide some analysis of the graph similarities provided by the generative model and explore whether they can be used for the purposes of clustering. One principled approach to this problem is to use the kernel principal component analysis explained in Section 5.5. In order to assess the quality of the method, we compare our embedding result with that obtained by using edit distance to measure graph dissimilarity. In Figure 5.5 , we illustrate the results of the Jensen-Shannon kernel embedding and edit distance embedding in the 2D space for two different object clustering tasks. The edit distance used is the approximate edit distance computed using the matchings from the

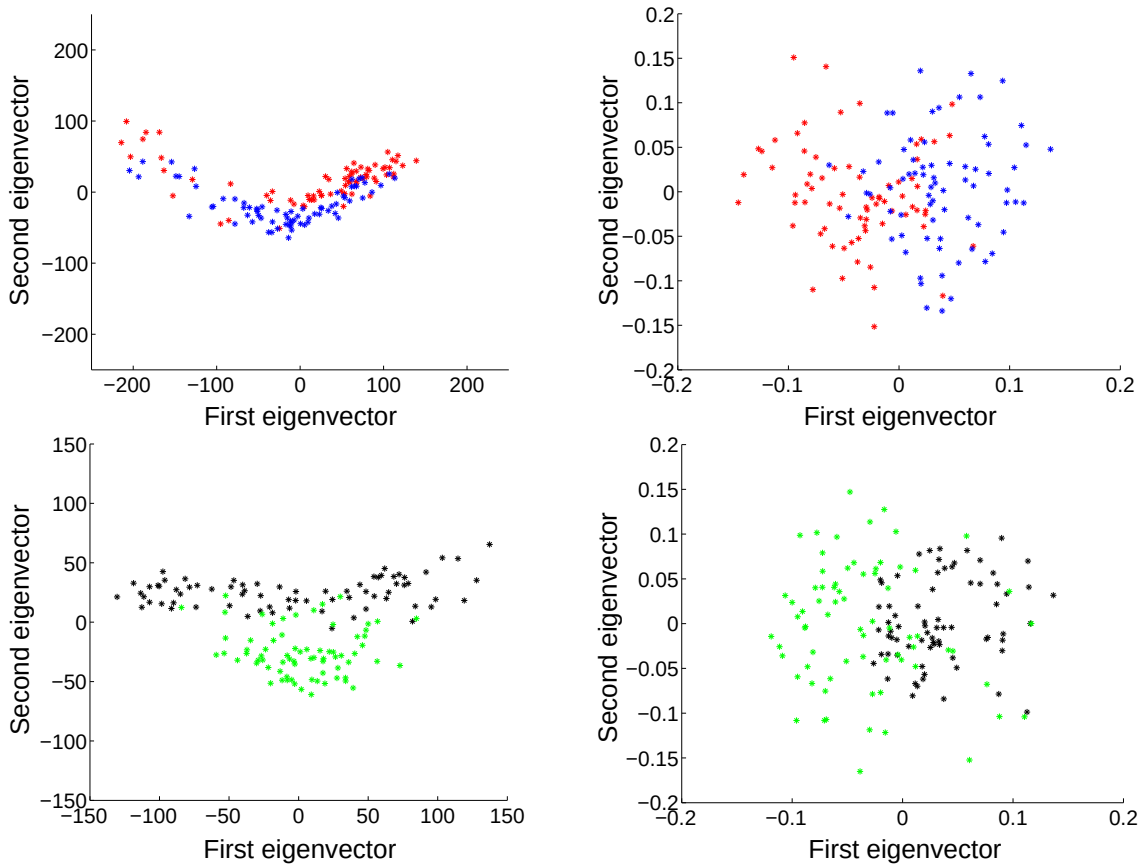


Figure 5.5: Comparison of graph clusterings obtained from Jensen-Shannon kernel and edit distance. Row 1: cat (red) and pig (blue). Row 2: bottle 1 (black) and bottle 2 (green). Column 1: edit distance and Column 2: Jensen-Shannon kernel.

graduated assignment [48]. The top row shows the embeddings of graphs from images of the cat (red) and the pig (blue) from the COIL dataset. The second row shows the embedding of the graphs from two types of bottle images (bottle1 as black scatter points and bottle2 as green scatter points) from the COIL dataset. The left hand column displays the clustering results obtained by edit distance and the right hand column gives the result obtained using the Jensen-Shannon kernel. To evaluate the quality of the clustering results obtained using the two methods, we measure the cluster compactness and separation using the Dunn index [32]. The Dunn index is defined as the ratio between the minimal

Table 5.2: The Dunn index for the clusterings obtained from the two different embeddings. The best results are shown in bold.

Dunn index	cat & pig	two bottles
Jensen-Shannon embedding	**0.7974**	**0.8309**
edit distance embedding	0.5217	0.5937

inter-cluster distance and the maximal intra-cluster distance. The higher the value of the index the better the separated clusters. We measure the inter-cluster distance between two clusters as the distance between their centroids (mean of the data points inside a cluster). The intra-cluster distance of a cluster is measured as the average distance of the data points inside the cluster to its centroid. Table 5.2 compares the Dunn index for the clusterings obtained from the two different embeddings. From Table 5.2, it is clear that the Jensen-Shannon embedding outperforms the edit distance embedding for both object clustering tasks.

5.7.4 Sampling New Graphs

Finally, we generate graphs using our method in Section 5.6 and explore to what extent the sample graphs from the generative model reproduce the statistical properties of the original graphs used to train the supergraph model. To do this, we experiment with both a synthetic graph dataset and Delaunay graphs from the real-world COIL image dataset. The synthetic dataset contains two types of representative graph models. The first are the classical Erdös-Rényi (ER) random-graphs [33]. These are constructed by connecting each pair of nodes in the graph with an equal probability p ($0 \leq p \leq 1$), independently of the other edges. The second class of graphs are Barabási and Albert (BA) scale-free networks whose node degree follows the power-law distribution shared by many real-world networks. The scale-free networks here are generated with the preferential attachment al-

gorithm [7]. This preferential attachment algorithm commences from a small seed graph of size m_0 and iteratively introduces one new node to the graph at a time, by connecting it to m ($1 \leq m \leq m_0$) existing nodes with a probability that is proportional to the degrees of the existing nodes. There are 40 ER graphs in the synthetic dataset and these graphs are constructed using a common value of p=0.1. The 40 BA scale-free graphs in the synthetic dataset are constructed from a same seed graph of size m_0=5 and using a common value of m=3. The number of the nodes in both types of graphs satisfies a Gaussian distribution $N(110, \sqrt{70})$ and vary from 90 to 130. The Delaunay graphs used here are from the 72 pig images in the COIL dataset.

We construct a generative model for each type of graph and sample graphs from the resulting generative models using the procedure given in Algorithm 1. We compare the following statistical properties of both the training graphs (the ones used to construct the supergraph) and the sample graphs from the generative model, i.e. 1) the node degree distribution , 2) the graph diameter distribution, 3) the distribution of relative frequency for paths of a chosen length l, here l =5, 4) a scatter plot of the Ihara coefficients of the graphs which count the number of the (prime cycles) triangles, squares and pentagons of graphs as feature vector [77], 5) the eigenvalue distribution for the normalized Laplacian matrix of graphs and 6) the distribution of a graph spectral characterization (the derivative of the Riemann zeta function at the origin [111]).

It is worth pausing to consider the challenges posed by simulating these different characteristics. First our model assumes neither a node frequency distribution nor a degree distribution. This is learned from the data. Second, we assume no detailed model of edge connectivity and this is again learned from the data. Hence by simulating the node degree distribution, we explore the ability of our method to learn this from data. Second, the attributes 2) to 6) explore in a deep way the accuracy of both the node degree and edge connectivity models learned from the data.

Figure 5.6 shows the plots of these statistics for the ER graphs and the sample graphs

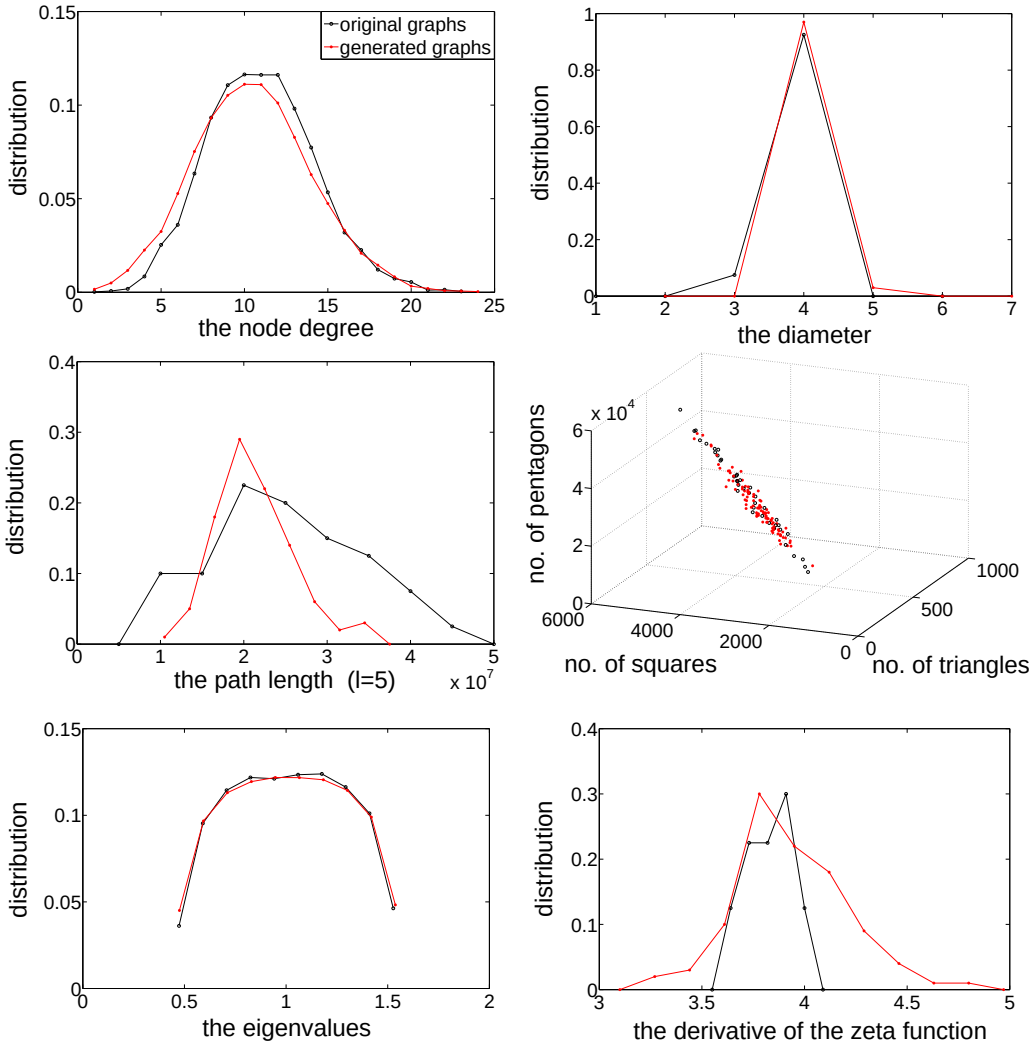


Figure 5.6: Comparison of the statistics for the ER graphs and their sample graphs.

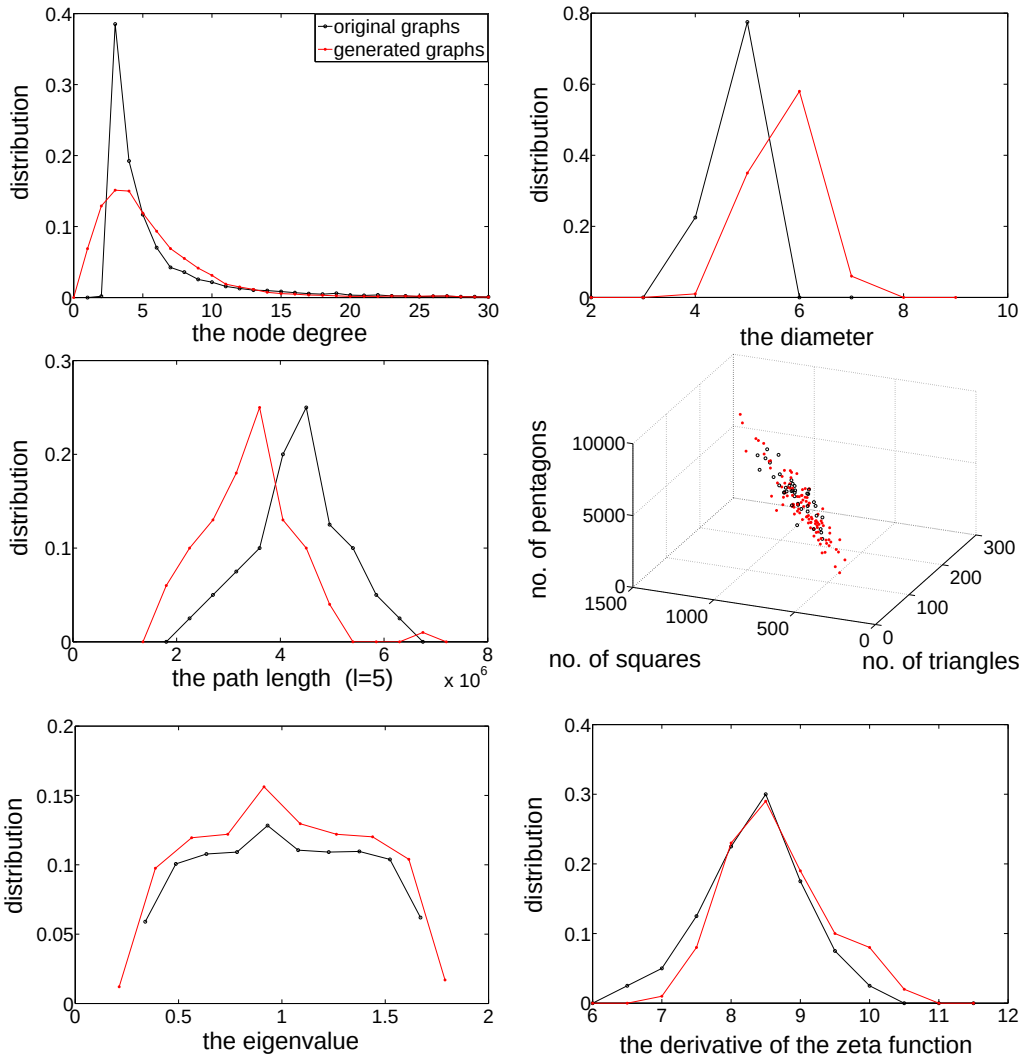


Figure 5.7: Comparison of the statistics for the BA scale-free graphs and their sample graphs.

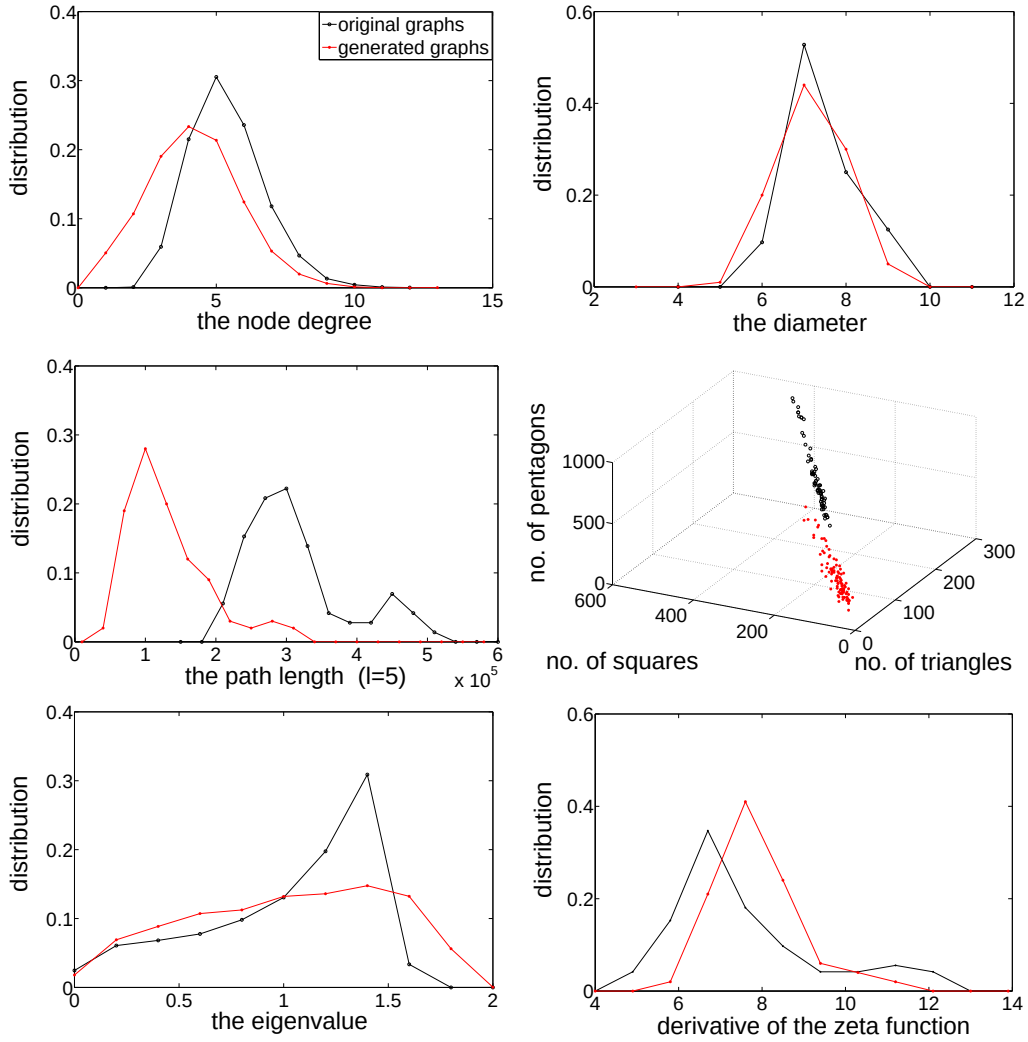


Figure 5.8: Comparison of the statistics for the Delaunay graphs and their sample graphs.

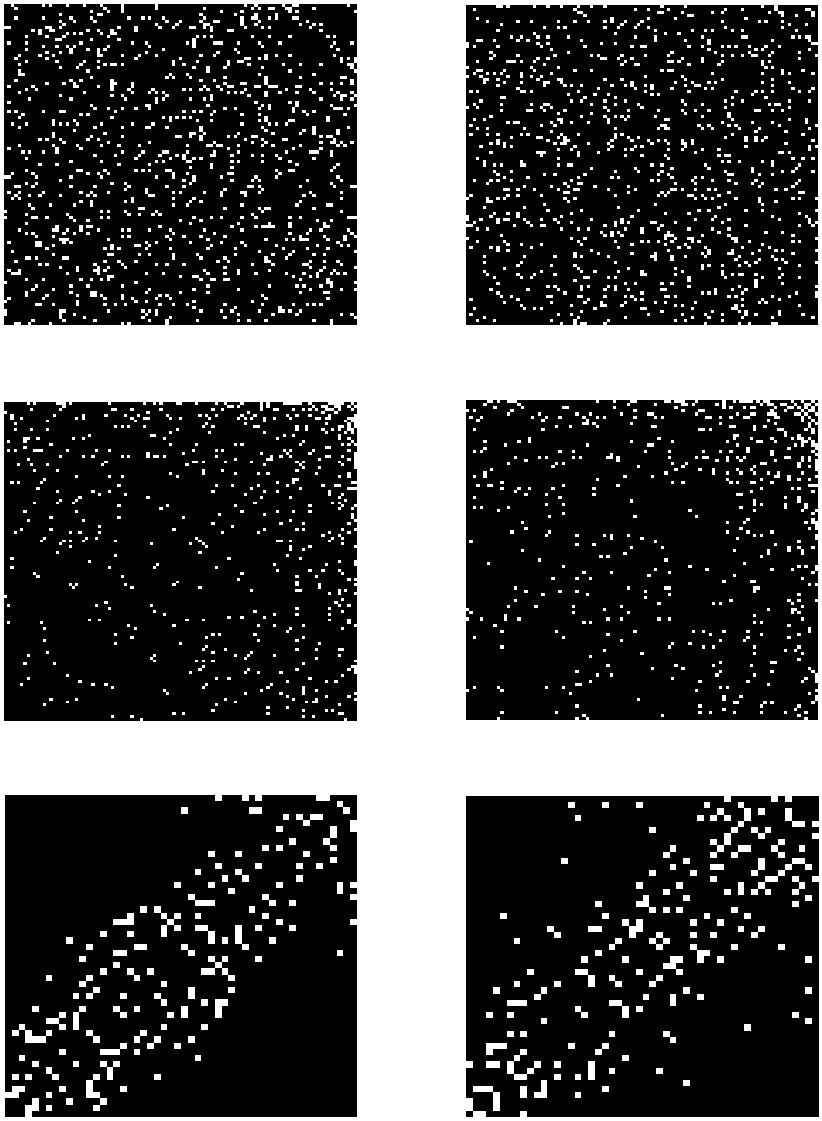


Figure 5.9: The adjacency matrices of some sample graphs where black and white squares are used to indicate zero and unit elements of the adjacency matrices. Top row: from ER supergraph. Middle row: from BA scale-free supergraph. Bottom row: from Delaunay supergraph.

from their corresponding generative model. We represent the results for the original graphs using black and those for the sample graphs using red. From these plots, we observe that the sample graphs reproduce the distributions of graph statistics of the ER graphs well. This is especially the case for the node degree distribution, the graph diameter distribution and the normalized Laplacian eigenvalue distribution where there are only slight deviations. The original graphs and the sample graphs have similar curves in the path length distribution and the distribution of the derivative of zeta function. Their Ihara coefficient scatter points are also overlapped. Figure 5.7 illustrates the distributions obtained for the BA graphs and their corresponding sample graphs. When plotting node degree distribution, although the curve for the sample graphs does not as peak in the same way as its counterpart, it still exhibits a similar increasing and decreasing pattern as the BA graphs. For the remaining distributions, the sample graphs give similar results to those of the original graphs. Compared to the results in Figure 5.6 and Figure 5.7, Figure 5.8 shows two significant deviations between the distributions of the Delaunay graphs and their sample graphs. One is in the Ihara coefficient scatter plot where the original graphs and the sample graphs display two separated clusters. The other is that the normalized Laplacian eigenvalue distribution of the sample graphs is more uniform than that for the original graphs.

From the plots in the three figures above, we observe that the properties of the sample graphs from the ER generative model resemble the original graphs most closely, and least well for the Delaunay graphs. The reason for this resides in the way we sample graphs, based on the assumption that the nodes and edges of graphs arise independently. For the three types of graphs studied, the ER graphs fit this model best. The Delaunay graphs which are constructed by triangulation violate the assumption most strongly. Nevertheless, for all the three types of graphs, the graphs sampled from the generative model by our method exhibit comparable properties to those of the original graphs to some extent.

In Figure 5.9, we visualize the adjacency matrices for some sample graphs for the

three different types of varieties of graphs studied. In the plots we use black and white squares to respectively represent zero and unit elements of the adjacency matrices. We note the sample graphs exhibit different edge densities and patterns of connectivity. The generated sample graphs from the ER supergraph demonstrate a uniform distribution of edges, whereas in the case of the BA supergraph there is condensation of edges around a few nodes. The edge density of the graphs sampled from the Delaunay supergraph is most unbalanced.

The overall conclusions of this study are that our method can learn and then generate distributions that reflect not only node degree statistics, but characteristics which are an artifact of detailed models of edge connectivity.

5.8 Conclusions

One big novelty of this chapter is that we have developed an information theoretic framework for learning a generative model (in the form of a supergraph) for graphs which captures the probabilistic distributions over nodes and over edges. We also have developed a novel practical algorithm for solving the problem. That is, we have provided a variant of the EM algorithm for estimating both the structure of the supergraph and node correspondences between the supergraph and the sample graphs. Empirical results on real-world datasets have shown the effectiveness of our proposed method. We have also illustrated how to embed graphs using supergraphs with Jensen-Shannon divergence and investigated the performance of our generative model on sampling new graphs. There are a number of ways in which the work reported here can be extended. First, since the probabilistic framework we used here is based on the edge connectivity of graphs, our work concentrates on unweighted graphs. There is scope for generalizing the method to weighted and attributed graphs. Second, the procedures of learning the structure of the generative model (i.e. edge connectivity) and its node and edge occurrence probabilities

are realized as discoupled computational procedures, yet they are clearly closely dependent. A better procedure will be to realize the estimation of the two parts jointly.

Chapter 6

Information Theoretic Prototype Selection for Graphs

In this chapter we present a prototype size selection method for a set of sample graphs. Our method of prototype size selection is based on the theory of approximate set coding. Approximate set coding was initially proposed for clustering validation in the vector domain, here we extend the theory from the vector domain to graph domain and apply it to selecting prototype graph size. However, extending the theory to graph domain is by no means a trivial problem due the difficulty of manipulating graph structures. Our main contributions here are that 1) we redefine the three critical concepts and reformulate the functions in approximate set coding so that the theory can be adopted for graphs, and 2) we solve the problem of exploring all the possible correspondence between the data graphs and prototype graphs by sampling the correspondence using the importance sampling approach. With the new definitions and the facility of the importance sampler in hand, we pose the problem of prototype size selection as that of optimizing the mutual information between two partitioned sets of sample graphs. In the experiments, we apply our method to the graphs from the COIL image dataset and investigate its performance on prototype size selection tasks.

6.1 Introduction

A problem we may encounter when dealing with graph data is to select the best prototype graph from several candidate prototype graphs in hand. This problem falls into the category of model selection, which is one of the fundamental tasks in pattern analysis. There are a wealth of principles in the literature for model selection [79][44][87]. Generally speaking, although these principles are motivated from different viewpoints, most of them employ penalizing the parameters (or complexity) of the model in order to generalize well on a new dataset. For instances, the two-part minimum description length criterion we adopted to construct our supergraph model in Chapter 5 involves penalizing the complexity of the model using the von Neumann entropy. Other examples also include the Akaike's information criterion (AIC) which penalizes the model by twice the number of free parameters of the model [1] [15], the Bayesian Information Criterion (BIC) which suggests a stronger penalty than AIC, i.e. number of model parameters times logarithm of the number of samples [87], and the universal coding in the minimum description length criterion [52].

Recently, Buhmann *et al.* [20] [21] have proposed an information theoretic principle called approximate set coding to estimate the generalization ability of the models from training to test data. The idea behind this can be explained using a communication protocol. The training data, after a transformation, generate a code for communication over a noisy channel and the test data recover the transformation after receiving the code. Channel capacity measures how well the communication between the two sets and models are ranked according to the channel capacity. The model that maximizes the channel capacity is selected. Actually, the channel capacity is encoded as the mutual information between the two sets. In their explanatory case of clustering model selection, both datasets in the scenario are characterized by a cost function and model selection is achieved by maximizing the channel capacity over a set of different cost functions.

Although these principles proposed for model selection mentioned above are widely

exploited in statistical models in the vector domain, their applications in the graph domain are very limited. This is due to the different representations between vectors and graphs. Vectors manifest themselves as ordered numerical values, while graphs are natural structures of edge and node (and also the attributes on them). Therefore, more effort is needed in order to adopt them in the graph domain. For instance, when we adopt the two-part minimum description length criterion to construct a supergraph for sample graphs in Chapter 5, graph characterizations from the von Neumann entropy are developed in advance to measure the complexity of the supergraph.

In this chapter we present an approach to selecting the optimal prototype graph size for a set of sample graphs. Our method is an extension of the theory of approximate set coding to the graphs. The prototype of optimal size is that which maximizes the mutual information between the two partitioned sets of the sample graphs. To measure the mutual information, we need to compute the partition functions of the two partitioned sets and their joint partition function. The computation of the partition function involves exploring the hypothesis space and this is a NP hard problem for graphs. We locate an approximate solution to this problem by using the importance sampling approach.

The remainder of the chapter is organized as follows. In Section 6.2 we first briefly introduce the idea of selecting prototype graphs using the theory of approximate set coding. In Section 6.3 we explain in detail how we extend the theory on model selection to the graph domain. This section includes four parts. The first three parts explain the new definitions of the three concepts (i.e. hypothesis, cost function, partition function) to cater for graph data. The last part shows how we approximate the value of the partition function using the importance sampling approach. In Section 6.4, we experiment with graph data to investigate our prototype size selection method. Finally, we conclude the work in this chapter in the last Section 6.5.

6.2 Approximate Set Coding

In this section we briefly introduce the idea of selecting prototype graphs using the theory of approximate set coding. In this context, a hypothesis is a solution to our pattern recognition problem. In this specific case, a hypothesis c is a mapping (matching) of all of our sample graphs to a prototype graph. We also have a cost function $R(c)$ which evaluates the quality of a particular matching. Naturally $R(c)$ depends on the prototype graph proposed for the data samples.

Given a prototype graph drawn from a set of possible prototypes (usually of different sizes or complexity), we can find the best matching and prototype configuration by optimizing $R(c)$. We denote the best hypothesis as $c^{\perp}$ that satisfies $c^{\perp} = \arg\min_{c} R(C)$. As usual, we cannot use $R(c)$ to select the best prototype from the set, as the more complex prototypes have lower costs (they fit the samples better) but do not generalize well.

In [20], Buhmann explains how the approximate set coding works for the clustering model selection problem by describing a communication scenario with a sender, a receiver and a problem generator where the problem generator serves as a noisy channel between the sender and receiver. In his theory, the communication between the sender and receiver take place in the noisy channel in the following procedures.

1. The sender and the receiver obtain a dataset that includes some objects to be clustered from the problem generator.

2. The sender and receiver calculate the number of hypotheses that are within a cost γ to the minimum cost of the clustering of the dataset.

3. The problem generator generates a new dataset and applies some transformations to the new dataset.

4. The problem generator sends the transformed dataset to the receiver without revealing the transformations.

5. The receiver calculates the hypotheses that are within a cost γ to the minimum cost of the clustering of the transformed dataset.

6. The receiver estimates the applied transformations by maximizing the overall number of hypotheses that are within a cost γ to the minimum cost of both datasets.

Approximate set coding uses the observation that there are a set of transformations which alter the sample data without essentially changing the prototype in any way. For example, if we consider the sample graphs in a different order, or if their nodes are permuted in some way, the structure of the recovered prototype should be the same (although the prototype graph nodes may also be in a different order). We can use this fact to measure how good our prototype is at recovering these transformations when they are coded using the prototype graph and sent through a noisy channel. To do this, we split the sample data into two partitions. The first partition is used to code the transformation, and the second provides a prototype graph to decode the transformation. We then attempt to maximize the amount of information transmitted. The analysis in [21] shows that the mutual information between sender and receiver is

$$I_\gamma = \frac{1}{N} \log \left(\frac{|\Omega||\Delta C_{\gamma,12}|}{|C_{\gamma,1}||C_{\gamma,2}|} \right), \tag{6.1}$$

where N is the number of graphs in the partitioned sets and $|\Omega|$ is the number of free transformations of the graphs. $|C_{\gamma,1}|$ is the number of hypotheses that are within a cost γ of the best cost in set 1 (and likewise for $|C_{\gamma,2}|$). The quantity $|\Delta C_{\gamma,12}|$ is the number of hypotheses on set 2 which are within a cost γ of the best cost in set 1. To calculate this, we need a way of transferring hypotheses from set 2 to set 1. In the following, we will describe in detail how we extend this theory to apply it to the graph prototype size selection.

6.3 Prototype Selection for Graphs

We commence by introducing our problem and then give formal definitions of the ingredients. Given a set of sample graphs, our aim is to select the optimal size of the prototype graph for the sample graphs. To ensure that the optimal prototype graph generalizes well on a new dataset, we adopt the two-sample scenario and partition the sample graphs into two sets of the same size $\mathcal{G}^{(1)} = \{G_1^{(1)}, \ldots, G_i^{(1)}, \ldots, G_N^{(1)}\}$, $\mathcal{G}^{(2)} = \{G_1^{(2)}, \ldots, G_i^{(2)}, \ldots, G_N^{(2)}\}$. Here the superscripts indicate different sample-sets and the subscripts indicate the graph indices. To partition the graphs from images of the same object into two sets, we index the graphs according to their image viewpoints and allocate neighbour graphs in the index to different sets. The best prototype graph is determined according to its generalization capability on the two sets.

6.3.1 Hypothesis

The hypotheses originally proposed in the clustering problem (where approximate set coding was first used) are the assignments of data points to clusters [20]. Here in our problem the hypotheses consist of a set of mappings of each of the sample graphs onto its corresponding prototype graph. By direct analogy with the clustering problem, each mapping is equivalent to an assignment of a point to a cluster; the prototype graph here is equivalent to the cluster centroid. For each dataset $\mathcal{G}^{(q)}$ $(q \in \{1, 2\})$ a hypothesis is $c_q = \{\mathbf{S}_1^{(q)}, \ldots, \mathbf{S}_i^{(q)}, ..., \mathbf{S}_N^{(q)}\}$ where $\mathbf{S}_i^{(q)}$ $(i \in \{1, 2, ..., N\})$ is the assignment matrix between graph $G_i^{(q)}$ from set $\mathcal{G}^{(q)}$ and its corresponding prototype graph $\Gamma^{(q)}$. The set of all possible hypotheses is $\mathcal{C}_q$, which consists of all the possible mappings between all samples and the prototype graph.

6.3.2 Cost Function

To proceed, we require a cost function $R_q(c_q)$ to quantify the effectiveness of a particular hypothesis c_q. The cost function measures how consistent the given mappings are with the prototype graph. Here the cost function of a hypothesis is the negative logarithm of the likelihood of the sample graphs from the prototype graph under the hypothesis, which uses the probabilistic framework presented in Chapter 3

$$\begin{aligned} R_q(c_q) &= -\ln P(\mathcal{G}^{(q)}|\Gamma^{(q)}, c_q) \\ &= -\sum_{G_i^{(q)}} \sum_{a \in V_i^{(q)}} \ln \sum_{a \in V_\Gamma^{(q)}} K_a^i \exp\left[\mu \sum_{b \in V_i^{(q)}} \sum_{\beta \in V_\Gamma^{(q)}} A_{iab}^{(q)} M_{\alpha\beta}^{(q)} S_{ib\beta}^{(q)}\right]. \end{aligned} \tag{6.2}$$

In the above, $\mathbf{A}_i^{(q)}$ is the adjacency matrix for the sample graph G_i from set q and $\mathbf{M}^{(q)}$ is the adjacency matrix for the prototype graph $\Gamma^{(q)}$. The matrix $\mathbf{S}_i^{(q)}$ is the assignment matrix between the two graphs. If nodes a and b of the sample graph $G_i^{(q)}$ are connected, their corresponding element $A_{iab}^{(q)}$ in $\mathbf{A}_i^{(q)}$ has a unit value otherwise it is zero. This is same for the adjacency matrix $M^{(q)}$ of the prototype graph $\Gamma^{(q)}$. The elements of the assignment matrix $S_{ia\alpha}^{(q)}$ are unit if node a in graph $G_i^{(q)}$ is matched to node α in graph $\Gamma^{(q)}$. The cost function above is a natural choice in our problem because it is also involved in measuring the likelihood of the sample graphs from the prototype graph during the learning procedure of the prototype graph.

In order to normalize the minimum cost of the hypotheses to zero, we define the relative cost of hypothesis. Suppose the optimal hypothesis (i.e. the hypothesis yielding the lowest cost between the sample graphs and their prototype graph) is $c_q^{\perp}$, the relative cost of the hypothesis c_q is $\Delta R_q(c_q) = R_q(c_q) - R_q(c_q^{\perp})$.

6.3.3 Partition Function

The measurement of the mutual information of the two sample sets requires counting the number of hypotheses $|C_{\gamma,1}|$ and $|C_{\gamma,2}|$ that are within a certain cost γ of the optimal solution. However, this is hard to do since it involves exploring all the hypotheses. Fortunately, this value can be estimated using concepts from statistical physics. Considering the hypotheses as microcanonical ensembles in statistical mechanics, their number can be estimated by calculating the partition function [20]

$$\mathcal{Z}_q = \sum_{c_q \in \mathcal{C}_q} \exp[-\varepsilon \Delta R_q(c_q)], \tag{6.3}$$

where ε is a positive scaling parameter known as the inverse computational temperature. Essentially, ε coarsens the precision of the partition function approximating the number of hypotheses that fit the sample set [21]. When ε is zero, the partition function is equal to the number of all the possible hypotheses. When ε is very large, the partition function only counts the number of optimal hypotheses. Because ε controls the number of hypotheses fitting the sample set, we will call these ε-optimal hypotheses. In our case, the hypothesis space is the set of all the possible mappings between the sample graphs and their prototype graph. The hypothesis space is very large and the computation of the partition function will be expensive. Later we show how we use the importance sampling approach to sample the mapping between the sample graphs and their prototype graph and approximate the value of the partition function.

To measure how well the hypotheses generalize for the two sample sets, we count the number of ε-optimal hypotheses in the first set which also exist in the second set, when transferred to the first set. We therefore need a way of transferring hypotheses from the second set to the first. We denote the cost of the hypothesis c_2 between the transferred graphs and prototype graph $\Gamma^{(2)}$ as $R_t(c_2)$. This is the cost of making hypothesis c_2 for the graphs $\mathcal{G}^{(2)}$ when evaluated against the data in $\mathcal{G}^{(1)}$. The following procedure may be

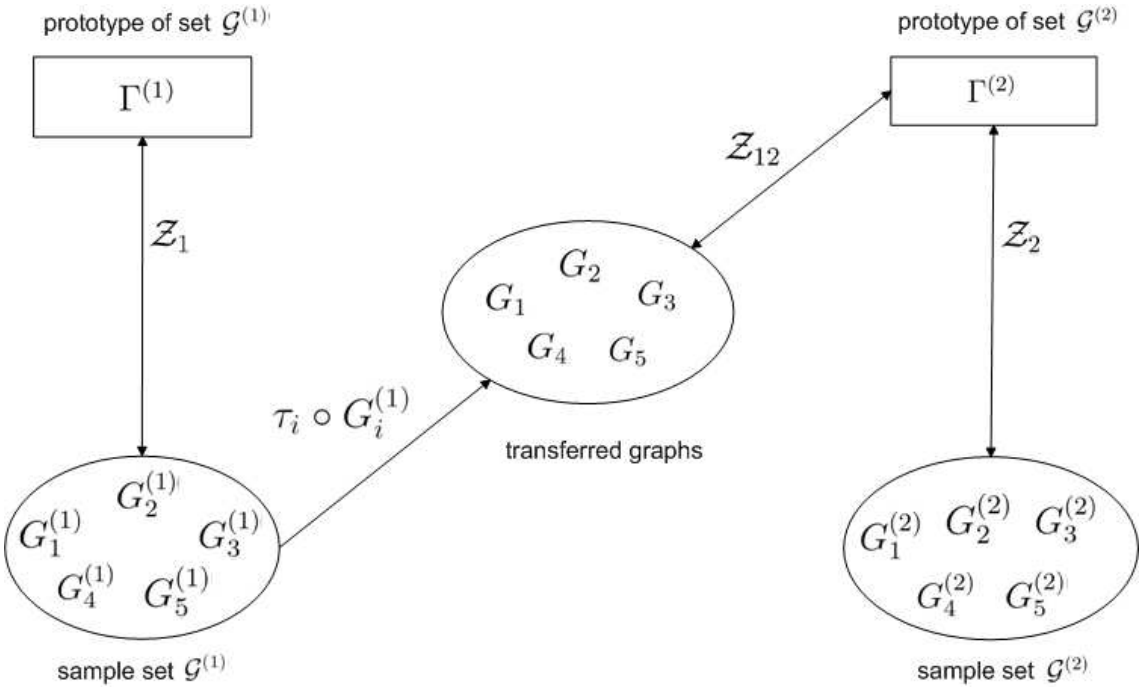


Figure 6.1: A diagram illustrates the procedure of computing the three partition functions. When we compute the partition function $\mathcal{Z}_{12}$, we need to count how many of our hypotheses are ε-optimal when we use the prototype from set 2 and the data graphs from set 1. We therefore need a way of transferring hypotheses from the second set to the first.

used to find the transfer. For each $G_i^{(1)}$ graph in $\mathcal{G}^{(1)}$, we find the most similar graph in $\mathcal{G}^{(2)}$ and the mapping τ_i between the two. $\tau_i \circ G_i^{(1)}$ is then the image of this graph in the second set. From these images, we compute the cost of c_2 by comparing the images to the prototype graph $\Gamma^{(2)}$ under the mappings in c_2. Finally, the joint partition function is formulated as

$$\mathcal{Z}_{12} = \sum_{c_2 \in \mathcal{C}_2} \exp[-\varepsilon(\Delta R_t(c_2) + \Delta R_2(c_2))] \,. \tag{6.4}$$

The quantity $\Delta R_t(c_2)$ is the relative cost of hypothesis c_2 between the image graphs of $\mathcal{G}^{(1)}$ in the second set and the prototype graph $\Gamma^{(2)}$. This is equivalent to the cost of hypothesis c_2 between the image graphs and $\Gamma^{(2)}$ minus their minimum cost. Figure 6.1 illustrates the procedure of computing partition functions $\mathcal{Z}_1$, $\mathcal{Z}_2$ and the joint partition function $\mathcal{Z}_{12}$.

Prototype graphs with different sizes are ranked according to their mutual information between the two sets

$$I_\varepsilon = \frac{1}{N} \log \left(\frac{|\Omega| \mathcal{Z}_{12}}{\mathcal{Z}_1 \mathcal{Z}_2} \right) . \tag{6.5}$$

In the above equation, $\mathcal{Z}_1$ and $\mathcal{Z}_2$ are respectively the partition functions of two sample sets, and $\mathcal{Z}_{12}$ is their joint partition function. $|\Omega|$ is the number of the free transformations of the graphs. In ideal conditions, its value is $|\Omega| = |V_\Gamma|!$, which is equal to the factorial of the size of the prototype graph. Since we are going to use the importance sampling approach to sample the correspondences in the hypothesis space rather than enumerating all the correspondences in the hypothesis space, and this will induce a bias on the value of $|\Omega|$. In practice we set its value to the one that keeps the value of the mutual information equal to zero when ε is zero. The amount of the mutual information can be interpreted as the generalization capacity of prototype graphs. Hence our problem is posed as that of finding the prototype graph that maximizes this mutual information.

6.3.4 Approximating the Partition Function

As previously mentioned, the computation of the partition function is expensive, since it involves exploring the hypothesis space, which encompasses all the mappings between the graphs in the sample sets to the prototype graph. To deal with this problem, we use the importance sampling approach proposed by Torsello [97] to sample the mappings from the hypothesis space and to approximate the value of the partition function.

Importance sampling [53] is a Monte Carlo sampling technique, where the expectation value of a particular distribution is approximated by a weighted average of random drawn from another distribution [95]. This technique is particularly useful to reduce the variance of the estimators. Suppose we aim to estimate the expectation value of a target function $g(x)$ in the domain $\mathcal{X}$, $E[g(x)] = \frac{1}{\|\mathcal{X}\|}\int_{\mathcal{X}} g(x)dx$. The basic idea of importance sampling is that instead of using random variables from $g(x)$, we use random variables from a different distribution $f(x)$ to estimate the expectation. Let $\mathbf{x} = (x_1, ..., x_k)$ be k random samples from the distribution $f(x)$. Thus we estimate $E[g(x)]$ as

$$E[g(x)] \approx \frac{1}{k}\sum_{i=1}^{k} g(x_i)\frac{\frac{1}{\|\mathcal{X}\|}}{f(x_i)} \ . \tag{6.6}$$

In our problem, we aim to approximate the value of the partition functions $\mathcal{Z}_q$ ($q \in \{1, 2\}$) and $\mathcal{Z}_{12}$. Since the approximation procedure is the same in all the three cases, we simply review the equations for $\mathcal{Z}_q$ ($q \in \{1, 2\}$). To commence, we have

$$\mathcal{Z}_q = \sum_{c_q \in \mathcal{C}_q} \exp[-\varepsilon \Delta R_q(c_q)] = E\Big[\exp[-\varepsilon \Delta R_q(c_q)]\Big]|\mathcal{C}_q|, \tag{6.7}$$

where $|\mathcal{C}_q|$ is the cardinality of the hypothesis space $\mathcal{C}_q$, in other words, the number of the mappings in $\mathcal{C}_q$. In this case, we have $\|\mathcal{X}\| = |\mathcal{C}_q|$ and $g(c_q) = \exp[-\varepsilon \Delta R_q(c_q)]$, and thus

$$E\Big[\exp[-\varepsilon \Delta R_q(c_q)]\Big] \approx \frac{1}{|\mathcal{C}'_q|}\sum_{c_q \in \mathcal{C}'_q} \exp[-\varepsilon \Delta R_q(c_q)]\frac{\frac{1}{|\mathcal{C}_q|}}{P(c_q)} \ , \tag{6.8}$$

where samples in $\mathcal{C}'_q$ are drawn from the distribution with a probability of $P(c_q)$.

Substituting Equation (6.8) into Equation (6.7), $\mathcal{Z}_q$ results in

$$\mathcal{Z}_q \approx \frac{1}{|\mathcal{C}'_q|} \sum_{c_q \in \mathcal{C}'_q} \frac{\exp[-\varepsilon \Delta R_q(c_q)]}{P(c_q)}. \tag{6.9}$$

Recall that $\Delta R_q(c_q) = R_q(c_q) - R_q(c_q^{\perp})$ and $R_q(c_q) = -\ln P(\mathcal{G}^{(q)}|\Gamma^{(q)}, c_q)$, where $\mathcal{G}^{(q)}$ is the observed graph and $\Gamma^{(q)}$ is the prototype graph. In order to estimate $\mathcal{Z}_q$, we need to sample hypotheses $c_q \in \mathcal{C}'_q$ with probability close to $\frac{P(\mathcal{G}^{(q)}|\Gamma^{(q)}, c_q)}{\sum_{c_q \in \mathcal{C}_q} P(\mathcal{G}^{(q)}|\Gamma^{(q)},\ c_q)}$. We assume that the graphs in the sample sets $\mathcal{G}^{(q)}$ are independent and sample mappings for individual graphs. The requisite for sampling a mapping between a graph and the prototype graph is a node-correspondence matrix, which gives the probabilities of the nodes in the graph corresponding to nodes in the prototype graph. This node-correspondence matrix can be obtained by performing a graph matching algorithm and by relaxing the resulting assignment matrix. The relaxing process ensures that there is a possibility that any node in the graph may be matched to any node in the prototype graph. The node-correspondence matrix obtained is a doubly-stochastic matrix, where the sum of each row and column is one. Once we have the node-correspondence matrix in hand, a mapping between the graph and the prototype graph can be located using the following procedure, as reported by Torsello [97].

Suppose the node-correspondence matrix is represented by $\bar{\mathbf{S}} = (\bar{s}_{a\alpha})$, which gives the probability that node a in the graph corresponds to node α in the prototype graph. We first sample a correspondence for the node indexed 1 in the prototype graph by picking a node a_1 in the graph, with probability $\bar{s}_{a_1 1}$. The next step is to condition the node-correspondence matrix to the current match by taking into account the structural information between the sampled node and all those remaining, which yielding a matrix $\bar{\mathbf{S}}^1_{a_1}$ that gives the conditional node-correspondence probability between the remaining nodes in the graph and those in the prototype graph given the current node cor-

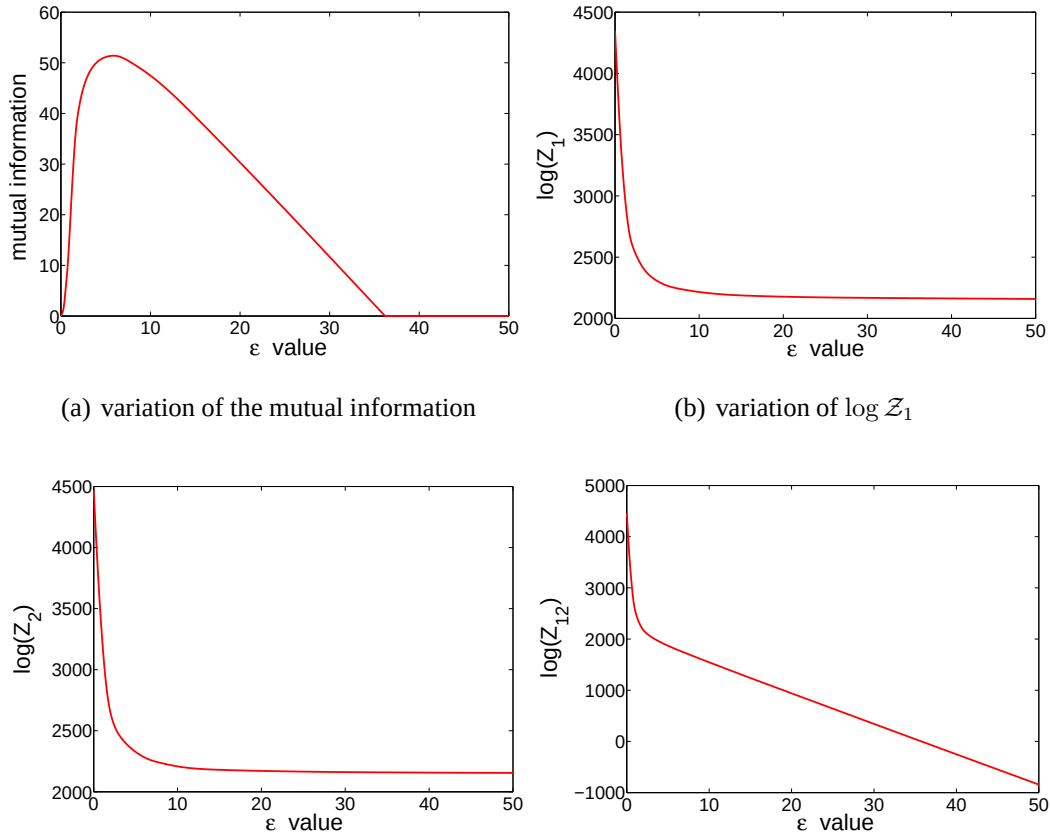
respondence. We proceed to sample a correspondence for the node indexed 2 in the prototype graph according to the matrix $\bar{S}^1_{a_1}$ and then compute a new conditional node-correspondence matrix. Iterating these steps until all the nodes in the prototype graph are matched to nodes in the graph. Finally, the probability of the sampled mapping $\breve{S}$ is $P(\breve{S}) = (\bar{s})_{a_1,1} \cdot (\bar{s}^1_{a_1})_{a_2,2} \cdot \ldots \cdot (\bar{s}^{1,\ldots,|V_\Gamma|-1}_{a_1,\ldots,a_{|V_\Gamma|-1}})_{a_{|V_\Gamma|},|V_\Gamma|}$. Sample a mapping $\breve{S}^{(q)}_i$ for each graph $G^{(q)}_i \in \mathcal{G}^{(q)}(i \in \{1,2,\ldots,N\})$ and these mappings constitute a hypothesis c_q, whose probability is $P(c_q) = \prod_{G^{(q)}_i \in \mathcal{G}^{(q)}} \breve{S}^{(q)}_i$.

6.4 Experiments

In this section, we report some experimental results of the application of our prototype size selection method on real-world dataset. The dataset used is the COIL [68] dataset.

We first investigate how the value of the mutual information and the three partition functions vary as the value of ε increases. To do this, we randomly partition the graphs from a given object, e.g. the cat images, into a training set and a test set that are of the same size. The bijective mapping of the graphs between the two sets is located by minimizing the sum of the approximate edit distances between the mapped graphs. The approximate edit distance is computed using the matchings from the graduated assignment [48]. We learn two prototype graphs of the same size for the two sets using the method in Chapter 3. Given this setting, we compute the value of the mutual information and the logarithms of the three partition functions $\log \mathcal{Z}_1$, $\log \mathcal{Z}_2$ and $\log \mathcal{Z}_{12}$.

Figure 6.2 shows how these quantities vary as we increase the value of ε from 0 to 50. From the plot in Figure 6.2(a), we observe that the mutual information initially increases and achieves the highest value around ε=8, and afterwards it begins to decrease. To maintain the non negativity of the mutual information, we set its value to zero when it falls below zero. Figure 6.2(b) and Figure 6.2(c) respectively show the value of the logarithms of partition functions $\log \mathcal{Z}_1$ and $\log \mathcal{Z}_2$. From the plots it is clear that these



(a) variation of the mutual information

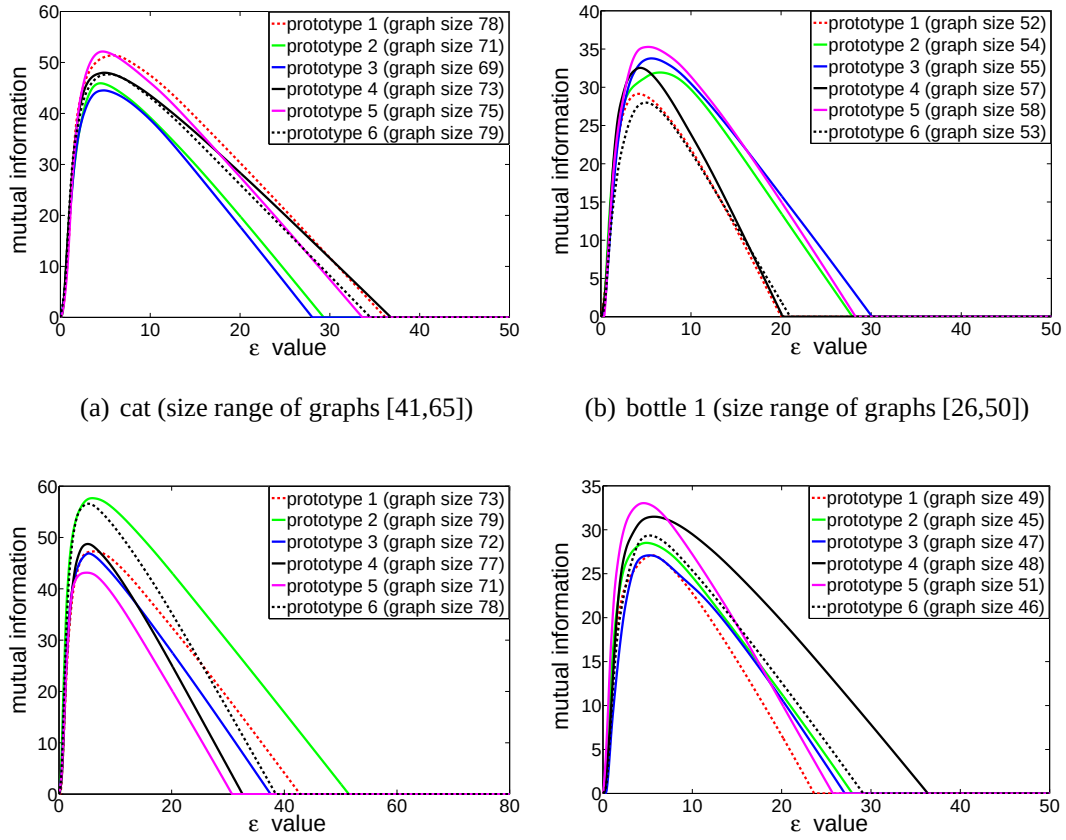
(b) variation of $\log \mathcal{Z}_1$

(c) variation of $\log \mathcal{Z}_2$

(d) variation of $\log \mathcal{Z}_{12}$

Figure 6.2: How the mutual information and the logarithm of partition functions change as ε increases from 0 to 50. (a) variation of the mutual information, (b) variation of $\log \mathcal{Z}_1$, (c) variation of $\log \mathcal{Z}_2$ and (d) variation of $\log \mathcal{Z}_{12}$.

two quantities converge to a horizontal asymptote. The reason for this is that the relative cost of the optimal hypothesis is zero and thus its contribution to the partition function is a constant positive value. The exponential of the relative costs given by the non-optimal hypotheses converges to zero as ε increases, thus yielding the observed horizontal asymptote. On the other hand, the logarithm of the joint partition function $\log \mathcal{Z}_{12}$ in Figure 6.2(d) continues to decrease as ε increases. This indicates that the optimal hypotheses of the graphs in the test set do not necessarily generalize to the optimal hypotheses of their mapped graphs in the training set. For this reason the relative costs of all the hypotheses



(a) cat (size range of graphs [41,65])

(b) bottle 1 (size range of graphs [26,50])

(c) pig (size range of graphs [44,70])

(d) bottle 2 (size range of graphs [32,44])

Figure 6.3: Variation of the mutual information of six prototype graphs of the four objects. (a) Cat object, (b) bottle 1 object, (c) pig object and (d) bottle 2 object.

in the joint partition function are positive values. As a result their exponentials converge to zero as ε increases. Consequently, the joint partition function converges to zero and its corresponding logarithm becomes both large and negative.

Our second experimental goal is to select the optimal sizes of the prototype graphs for several objects from the COIL dataset. Here the objects we used are the cat, pig and two bottles. To perform these tasks, for each object we learn six prototype graphs of different sizes using the method in Chapter 3 and then compute the mutual information of these prototype graphs. The optimal size of the prototype graph is that which gives the highest mutual information as ε varies. Figure 6.3 shows plots of the mutual information

Table 6.1: The sizes of the six prototype graphs. The sizes of the prototype graphs selected are shown in bold.

Prototype Graph	cat	bottle 1	pig	bottle 2
prototype 1	78	52	73	49
prototype 2	71	54	**79**	45
prototype 3	69	55	72	47
prototype 4	73	57	77	48
prototype 5	**75**	**58**	71	**51**
prototype 6	79	53	78	46

of the six prototype graphs versus the value of ε for the four objects. The sizes of the six prototype graphs are shown on the legend and the size ranges of the graphs used to learn the prototype graphs are given following the names of the objects. From the plots it is clear that for each object there is a prototype size that gives optimized performance. In Table 6.1, we also list the sizes of the six prototype graphs, i.e. the number of the nodes in the prototype graphs, of the four objects. The sizes of prototype graphs selected by our model selection method are shown in bold. Note that unlike what is expected using other standard model complexity selection methods, which may choose the model with the smallest size, our experiments observe that in three out of four objects the proposed method favours the larger size.

6.5 Conclusions

In this chapter we have developed a method for selecting the optimal size of a prototype graph used to represent a set of sample graphs. Our method of prototype size selection is based on the theory of approximate set coding that was initially proposed for cluster-

ing validation in the vector domain. The main novelty of this chapter is that we redefine the three critical concepts in the theory of the approximate set coding and extend the theory from vector domain to graph domain so that we can apply the theory to solving model selection problems in graph domain. The second novelty of this chapter is that we have solved the problem of exploring all the possible correspondence between the sample graphs and prototype graphs by sampling the correspondence using the importance sampling approach. With the new definitions and the facility of the importance sampler in hand, we posed the problem of prototype size selection as that of optimizing the mutual information between two partitioned sets of sample graphs. In the experimental part, we have investigated its performance on prototype graph selection in object recognition. However, the method we presented in this chapter is a follow-up work after we have learned the structure of the prototype graphs. Therefore, learning the prototype graphs and selecting the prototype size are realized as discoupled computational procedures. In the future work, we will adopt some more sophisticated strategies (e.g. simulated annealing) to realize the estimation of the two parts jointly.

Chapter 7

Conclusions and Future Work

In this chapter we first provide a summary of the main contributions of the thesis. This includes the novel ideas on developing graph characterizations, constructing generative models and selecting prototype graphs. Secondly, we will spell out some of the weaknesses and describe possible directions for future work.

7.1 Summary of Contributions

We have developed and evaluated new methods for characterizing graphs and constructing generative models for graph data. Our generative models developed concentrate on capturing the variations of edge connectivity present in the sample graphs. We now provide a summary of our contributions for each chapter in the thesis.

7.1.1 A Supergraph-based Generative Model

Our first contribution is that we developed a novel generative model for a set of graphs based on a supergraph structure in Chapter 3. The supergraph is analogous to the graph union that aims to capture the structural variation present in the set. We began by introducing the *a posteriori* probability defined in a graph matching problem [61]. In the

subsequent development, we used this probability to measure the likelihood of a sample graph from the supergraph. The supergraph we aim to learn is one which maximizes the likelihood of the sample graphs. To maximize this objective function, the unknown correspondence information between the nodes of the sample graphs and those of the supergraph was treated as missing data and we developed a variant of the expectation-maximization (EM) algorithm to locate the supergraph structure. This supergraph can generate new graphs by modelling the edge occurrence probabilities. Besides, we also investigated the use of the von Neumann entropy as the indicator for measuring the complexity of the supergraph in the experimental part of this chapter.

7.1.2 Graph Characterizations from von Neumann Entropy

The second contribution of the thesis is that we developed graph characterizations from the von Neumann entropy. We first explored how the von Neumann entropy of a graph associated with the normalized Laplacian matrix can be used as a measure of graph characterization. Then we developed a simplified form for the von Neumann entropy of a graph that can be computed in terms of node degree statistics. The simplified form of the von Neumann entropy offers the advantage of lower computational complexity which is quadratic in the number of the nodes of graphs while the computation of von Neumann entropy is cubic. Both of the two measures belong to the invariant graph characterizations. We also compared the resulting characterizations with a number of alternative graph characterizations including Estrada's heterogeneity index [37] and the derivative of Riemann zeta function at the origin [111]. In the case of Estrada's heterogeneity index we revealed a new link between Estrada's index and the commute time on a graph. In addition, we also explored how the von Neumann entropy can be used in conjunction with the thermodynamic depth and illustrated its applications to biological networks.

7.1.3 An Information Theoretic Generative Graph Prototype

In this chapter, we combined the graph characterizations from the von Neumann entropy with the probabilistic framework described in Chapter 3, to construct a generative prototype for a set of graphs by adopting a minimum description length approach. Again here the generative graph prototype is represented by a supergraph structure. The complexity of the supergraph is encoded using the simplified von Neumann entropy. A variant of the EM algorithm is developed to minimize the overall description length in which both the structure of the supergraph and the node correspondences between the sample graphs and the supergraph are treated as missing data. We also exploited a kernel method of analyzing graph similarity. To do this, we measured graph similarities using the Jensen-Shannon divergence and then embedded graphs into pattern space using kernel principal component analysis. The Jensen-Shannon divergence between a pair of graphs is found by taking the difference between the entropy of the pairwise supergraph and the average of the separate entropies of the two graphs used to construct it. In addition, we also developed a method of generating new graphs from the supergraph. This is realized by assuming that both the nodes and edges of graphs arise under independent Bernoulli distributions and sampling new graphs according to their node and edge occurrence probabilities. Therefore, our supergraph model proposed in this chapter can fulfil the tasks of graph classification, graph clustering and of generating new graphs.

7.1.4 Information Theoretic Prototype Selection for Graphs

In Chapter 6, we provided a prototype graph size selection method. Our method of prototype size selection is based on the theory of approximate set coding that was initially proposed for clustering validation in the vector domain. We extended the theory from the vector domain to graph domain so that it can be applied to the model selection problem for graphs. However, extending the theory to graph domain is not a trivial problem. Our

main contributions in this chapter are that 1) we redefined the three critical concepts and reformulated the functions in approximate set coding so that the theory can be adopted for graphs, 2) we solved the problem of exploring all the possible correspondence between the sample graphs and prototype graphs by sampling the correspondence using the importance sampling approach. With the new definitions and the facility of the importance sampler in hand, we posed the problem of prototype size selection as that of optimizing the mutual information between two partitioned sets of sample graphs. Experimental investigations demonstrated the practical utility of our method.

7.2 Weaknesses

There are a number of weaknesses of the work presented in the thesis. We discuss these weaknesses and then propose some possible directions for future work.

In the methods for constructing generative models presented in Chapter 3 and Chapter 5, we developed variants of the EM algorithm to optimize the objective functions (which were respectively the likelihood of the sample graphs in Chapter 3 and the overall description length in Chapter 5) . The reason we use the EM algorithms is that the correspondence information between the nodes of the sample graphs and the supergraph is hidden to us, and the EM algorithms are specialized to solve the problems where there is missing data. Although we showed that our variants of the EM algorithm work well to drive the objective functions to converge, the supergraphs we obtained from these algorithms are not guaranteed to be the best solutions due to the fact that the EM algorithm can easily get stuck in local optima. In addition, because the probabilistic framework we adopted to learn the supergraphs (i.e. the likelihood function described in Section 3.2, Chapter 3) is developed in the context of unweighted graphs, the generative model construction methods we proposed in the thesis are restricted to unweighted graphs. Another weakness of the generative models proposed in the thesis is that they cannot generate graphs bigger

than the supergraphs. This is because we only model the occurrence probability of the nodes and edges on the supergraphs and sample nodes and edges already existing in the supergraphs to assemble new graphs.

In Chapter 4, we extracted two graph characterizations from the von Neumann entropy. However, both of the two representations, i.e. the von Neumann entropy of a graph and the simplified von Neumann entropy, have shortcomings. For the von Neumman entropy of a graph, since it is defined using the eigenvalues of the normalized Laplacian matrix, it suffers from the problem of cospectrality of graphs. Cospectral graphs have the same eigenvalues with respect to the matrix representation being used. Therefore, even for two graphs with different structures, the value of their von Neumann entropy may be same. For the simplified form of the von Neumann entropy, it is developed by using two equivalent transformations, i.e. the trace of the normalized Laplacian matrix of a graph is equal to the number of the nodes in the graph (refer to Equation 4.8) and the trace of the square of the normalized Laplacian is equal to a quantity of node degree statistics (refer to Equation 4.9). These equivalent transformations hold only for unweighted graphs. Thus, the simplification of the von Neumann entropy of a graph described here does not exhibit itself with the capability of handling edge-weighted graphs.

Additionally, we showed a prototype size selection method in Chapter 6. This method only deals with the problem of selecting the best prototype size from candidate prototypes and it does not involve the learning procedure of the prototype graphs. That is to say, the prototype selection method presented in the chapter is a separate post-processing step that takes place after the learning procedure of the prototype graphs. We need to carry out a further investigation on how to integrate the learning procedure and selecting procedure together so as to reduce the overall complexity.

7.3 Future Work

To address the weaknesses of this thesis, in this section we suggest some possible approaches to overcoming some of them for further research, and also provide a number of ways in which the work reported can be extended.

First, our work presented in the thesis solves the problem of constructing generative models for non-attributed graphs. There is scope for generalizing the methods to attributed graphs. Since the probabilistic framework we used to develop our generative models is based on non-attributed graphs, our methods of constructing generative models are restricted to non-attributed graphs. One possible way of applying our work for attributed graphs would be to adjust the current probabilistic framework. This may involve adding extra parameters to the probabilistic framework to model the attribute of nodes and edges. To accommodate the new probabilistic framework, the learning procedure will change accordingly. For instance, during the iterations of the EM algorithm, we need to re-estimate not only the structure of the supergraph but also the attribute parameters.

Second, the problem that our generative models cannot generate graphs bigger than the supergraphs might be solved by padding extra nodes and edges to the supergraphs. We could use these extra nodes and edges to model the occurrence of the nodes and edges apart from the ones in the supergraphs.

In Chapter 4, we described how we developed graph characterizations using the von Neumann entropy of graphs. These simple measures of graph entropy open up a number of interesting potential information theoretic avenues. These include their use as model complexity measures in the learning of generative models using a minimum description length approach, and their use in the construction of information theoretic kernels using the Shannon-Jensen divergence (which are shown in Chapter 5). Moreover, it would be interesting to explore whether the von Neumann entropy can be extended to more complex matrix representations including those for edge-weighted graphs, attributed graphs or hypergraphs.

For the prototype size selection method proposed in Chapter 6, learning the structure of the prototype graphs and selecting the prototype size are realized as discoupled computational procedures. A better way would be to adopt some more sophisticated strategies (e.g. simulated annealing) to realize the estimation of the two parts jointly.

Bibliography

[1] H. Akaike. A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, 19:716–723, 1974.

[2] K. Anand, G. Bianconi, and S. Severini. Shannon and von Neumann entropy of random networks with heterogeneous expected degree. *Physical Review E*, 83(3):036109, 2011.

[3] O. E. Badawy and M. Kamel. Shape representation using concavity graphs. *In Proceedings of the International Conference on Pattern Recognition*, pages 461–464, 2002.

[4] A. D. Bagdanov and M. Worring. First order gaussian graphs for efficient structure classification. *Pattern Recognition*, 36:1311–1324, 2003.

[5] X. Bai and E. R. Hancock. Heat kernels, manifolds and graph embedding. *In Proceedings of Joint IAPR International Workshops on Structural and Syntactic Pattern Recognition and Statistical Techniques in Pattern Recognition*, pages 198–206, 2004.

[6] X. Bai and E. R. Hancock. Graph clustering using heat content invariants. *In Proceedings of the Iberian Conference on Pattern Recognition and Image Analysis*, pages 123–130, 2005.

[7] A. L. Barabási and R. Albert. Emergence of scaling in random networks. *Science*, 286:509–512, 1999.

[8] H. G. Barrow and R. M. Burstall. Subgraph isomorphism, matching relational structures and maximal cliques. *Information Processing Letters*, 4:83–84, 1976.

[9] F. K. Bell. A note on the irregularity of graphs. *Linear Algebra and Its Applications*, 161(1):45–54, 1992.

[10] I. Bengtsson and K. Zyczkowski. *Geometry of Quantum States:An Introduction to Quantum Entanglement*. Cambridge University Press, 2006.

[11] Y. Berchenko and M. Teicher. Graph embedding through random walk for shortest paths problems. *In Proceedings of the Symposium on Stochastic Algorithms, Foundations and Applications*, pages 127–140, 2009.

[12] Dennis S. Bernstein. *Matrix Mathematics: Theory, Facts and Formulas*. Princeton University Press, 2 edition, 2009.

[13] N. L. Biggs. *Algebraic graph theory*. Cambridge University Press, 1993.

[14] C. Bishop. *Neural Networks for Pattern Recognition*. Oxford: Clarendon Press, 1995.

[15] H. Bozdogan. Model selection and akaike's information criterion (aic): the general theory and its analytical extensions. *Psychometrika*, 52(3):345–370, 1987.

[16] S. Braunstein, S. Ghosh, and S. Severini. The laplacian of a graph as a density matrix: a basic combinatorial approach to separability of mixed states. *Annals of Combinatorics*, 10(3):291–317, 2006.

[17] L. M. Bregman. The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming. *USSR Computational Mathematics and Mathematical Physics*, 7:200–217, 1967.

[18] J. S. Bridle. Training stochastic model recognition algorithms as networks can lead to maximum mutual information estimation of parameters. *In Proceedings of Advances in Neural Information Processing Systems 2*, pages 211–217, 1990.

[19] H. J. Briegel and R. Raussendorf. Persistent entanglement in arrays of interacting particles. *Physical Review Letters*, 86(5):910–913, 2001.

[20] J. M. Buhmann. Information theoretic model validation for clustering. *In proceedings of the IEEE International Symposium on Information Theory*, pages 1398–1402, 2010.

[21] J. M. Buhmann, M. H. Chehreghani, M. Frank, and A. P. Streich. Information theoretic model selection for pattern analysis. *In Proceedings of the JMLR Workshop on Unsupervised and Transfer Learning*, 7:1–15, 2011.

[22] W. L. Buntine. A guide to the literature on learning probabilistic networks from data. *IEEE Transactions on Knowledge and Data Engineering*, 8(2):195–210, 1996.

[23] J. Burbea and C. Rao. On the convexity of some divergence measures based on entropy functions. *IEEE Transactions on Information Theory*, 28(3):489–495, 1982.

[24] T. Calders, C. W. Gnther, M. Pechenizkiy, and A. Rozinat. Using minimum description length for process mining. *In Proceedings of the Annual ACM Symposium on Applied Computing*, pages 1451–1455, 2009.

[25] W. J. Christmas, J. Kittler, and M. Petrou. Structural matching in computer vision using probabilistic relaxation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(8):749–764, 1995.

[26] F. R. K. Chung. *Spectral graph theory*. American Mathematical Society, 1997.

[27] L. Collatz and U. Sinogowitz. Spektren endlicher grafen. *Abhandlungen aus dem Mathematischen Seminar der Universitšt Hamburg*, 21:63–77, 1957.

[28] R. Collobert and J. Weston. A unified architecture for natural language processing: deep neural networks with multi task learning. *In Proceedings of the International Conference on Machine Learning*, pages 160–167, 2008.

[29] T. F. Cootes, G. J. Edwards, and C. J. Taylor. Active appearance models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(23):681–685, 2001.

[30] T. Cover and J. Thomas. *Elements of Information Theory*. New York:John Wiley&Sons, 1991.

[31] B. Delaunay. Sur la sphere vide. *Izvestia Akademii Nauk SSSR, Otdelenie Matematicheskikh i Estestvennykh Nauk*, 7:793–800, 1934.

[32] J. Dunn. Well separated clusters and optimal fuzzy partitions. *Journal of Cybernetics*, 4:95–104, 1974.

[33] P. Erdös and A. Rényi. On the evolution of random graphs. *Publicationes Mathematicae*, 5:17–61, 1960.

[34] F. Escolano, E. R. Hancock, and M. A. Lozano. Birkhoff polytopes, heat kernels and graph complexity. *In Proceedings of the International Conference on Pattern Recognition*, pages 1–5, 2008.

[35] F. Escolano, M. A. Lozano, and E. R. Hancock. Heat flow-thermodynamic depth complexity in networks. *In Proceedings of the International Conference on Pattern Recognition*, pages 1578–1581, 2010.

[36] F. Escolano, M. A. Lozano, E. R. Hancock, and D. Giorgi. What is the complexity of a network? the heat flow-thermodynamic depth approach. *In Proceedings of the Joint IAPR International Workshops on Structural and Syntactic Pattern Recognition and Statistical Techniques in Pattern Recognition*, pages 286–295, 2010.

[37] E. Estrada. Quantifying network heterogeneity. *Physical Review E*, 82(6):066102, 2010.

[38] E. Estrada, M. Fox, D. Higham, and G. L. Oppo. *Network science: complexity in nature and techenology*. Springer, 2010.

[39] D. Feldman and J. Crutchfield. Measures of statistical complexity: why? *Physics Letters A*, 238:244–252, 1998.

[40] P. F. Felzenszwalb and D. P. Huttenlocher. Efficient graph-based image segmentation. *International Journal of Computer Vision*, 59(2):167–181, 2004.

[41] M. Ferrer, F. Serratosa, and E. Valveny. On the relation between the median and the maximum common subgraph of a set of graphs. *In Proceedings of the IAPR-TC-15 Workshop on Graph-based Representations in Pattern Recognition*, pages 351–360, 2007.

[42] M. Ferrer, E. Valveny, F. Serratosa, and H. Bunke. Exact median graph computation via graph embedding. *In Proceedings of the Joint IAPR International Workshops on Structural and Syntactic Pattern Recognition and Statistical Techniques in Pattern Recognition*, pages 15–24, 2008.

[43] M. Fischler and R. Elschlager. The representation and matching of pictorical structures. *IEEE Transactions on Computers*, 22(1):67–92, 1973.

[44] D. P. Foster and E. I. George. The risk inflation criterion for multiple regression. *Annals of Statistics*, 22:1947–1975, 1994.

[45] N. Friedman and D. Koller. Bayesian about network structure: a Bayesian approach to structure discovery in Bayesian networks. *Machine Learning*, 50:95–125, 2003.

[46] N. Friedman, M. Linial, I. Nachman, and D. Pe'er. Using Bayesian networks to analyze expression data. *Journal of Computational Biology*, 7(3–4):601–620, 2000.

[47] K. R. Gabriel and R.R. Sokal. A new statistical approach to geographic variation analysis. *Systematic Zoology*, 18:259–278, 1969.

[48] S. Gold and A. Rangarajan. A graduated assignment algorithm for graph matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(4):377–388, 1996.

[49] P. Gopalakrishnan, D. Kanevsky, A. Nadas, and D. Nahamoo. An inequality for rational functions with applications to some statistical estimation problems. *IEEE Transactions on Information Theory*, 37(1):107–113, 1991.

[50] L. Grady. Random walks for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(11):1768–1783, 2006.

[51] P. D. Grunwald. *Advances in Minimum Description Length*. MIT Press, 2005.

[52] P.D. Grunwald, I.J. Myung, and M.A. Pitt, editors. *Advances in Minimum Description Length Theory and Applications*. MIT Press, 2005.

[53] J. M. Hammersley and D. C. Handscomb. *Monte carlo methods*. John Wiley & Sons, New York, 1964.

[54] C. Harris and M. Stephens. A combined corner and edge detector. *In Proceedings of the Fourth Alvey Vision Conference*, pages 147–152, 1988.

[55] G. E. Hinton, S. Osindero, and Y. W. Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18:1527–1554, 2006.

[56] X. Jiang, A. Mnger, and H. Bunke. On median graphs: Properties, algorithms, and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1144–1151, 2001.

[57] B. Krishnapuram, C. M. Bishop, and M. Szummer. Generative models and Bayesian model comparison for shape recognition. *In proceedings of the International Workshop on Frontiers in Handwriting Recognition*, pages 20–25, 2004.

[58] X. Li and Y. Shi. A survey on the randic index. *MATCH: Communication in Mathematical and in Computer Chemistry*, 59:127–156, 2008.

[59] J. Lin. Divergence measures based on shannon entropy. *IEEE Transactions on Information Theory*, 37(1):145–151, 1991.

[60] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 99(2):91–110, 2004.

[61] B. Luo and E. R. Hancock. Structural graph matching using the em algorithm and singular value decomposition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(10):1120–1136, 2001.

[62] B. Luo, R. C. Wilson, and E. R. Hancock. Spectral embedding of graphs. *Pattern Recognition*, 36:2213–2230, 2003.

[63] B. Luo, R. C. Wilson, and E. R. Hancock. A linear generative model for graph structure. *In Proceedings of the IAPR international conference on Graph-Based Representations in Pattern Recognition*, pages 54–62, 2005.

[64] B. Luo, R. C. Wilson, and E. R. Hancock. A spectral approach to learning structural variations in graphs. *Pattern Recognition*, 39(6):1188–1198, 2006.

[65] M. Maier, U. von Luxburg, and M. Hein. Influence of graph construction on graph-based clustering measures. *In Proceedings of Advances in Neural Information Processing Systems*, pages 1–9, 2010.

[66] A. F. T. Martins, M. A. T. Figueiredo, P. M. Q. Aguiar, N. A. Smith, and E. P. Xing. Nonextensive information theoretic kernels on measures. *Journal of Machine Learning Research*, 10:935–975, 2009.

[67] B. D. McKay. Spanning trees in regular graphs. *European Journal of Combinatorics*, 4:149–160, 1983.

[68] S. A. Nene, S. K. Nayar, and H. Murase. Columbia object image library. *Columbia University*, 1996.

[69] R. Nock and F. Nielsen. Fitting the smallest enclosing bregman ball. *In Proceedings of the European Conference on Machine Learning*, pages 649–656, 2005.

[70] F. Passerini and S. Severini. The von Neumann entropy of networks. *International Journal of Agent Technologies and Systems*, 1:58–67, 2008.

[71] J. P. W. Pluim, J. B. A. Maintz, and M. A. Viergever. Image registration by maximization of combined mutual information and gradient information. *IEEE Transactions on Medical Imaging*, 19(8):809–814, 2000.

[72] F. P. Preparata and M. I. Shamos. *Computational Geometry: An Introduction*. Springer Verlag, 1985.

[73] H. J. Qiu and E. R. Hancock. Graph matching and clustering using spectral partitions. *Pattern Recognition*, 39:22–34, 2006.

[74] H. J. Qiu and E. R. Hancock. Clustering and embedding using commute times. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(11):1873–1890, 2007.

[75] M. Randić. Characterization of molecular branching. *Journal of the American Chemical Society*, 97:6609–6615, 1975.

[76] M. A. Ranzato, J. Susskind, V. Mnih, and G. Hinton. On deep generative models with applications to recognition. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2857–2864, 2011.

[77] P. Ren, R.C. Wilson, and E.R. Hancock. Graph characterization via ihara coefficients. *IEEE Transactions on Neural Networks*, 22(2):233–245, 2011.

[78] B. D. Ripley. *Pattern Recognition and Neural Networks*. New York:Cambridge University Press, 1996.

[79] J. Rissanen. Modelling by shortest data description. *Automatica*, 14:465–471, 1978.

[80] J. Rissanen. A universal prior for integers and estimation by minimum description length. *Annals of Statistics*, 11(2):419–431, 1983.

[81] J. Rissanen. Universal coding, information, prediction and estimation. *IEEE Transactions on Information Theory*, 30(4):629–636, 1984.

[82] J. Rissanen. *Stochastic complexity in statistical inquiry*. Singapore:World Scientific, 1989.

[83] A. Robles-Kelly and E. R. Hancock. A Riemannian approach to graph embedding. *Pattern Recognition*, 40(3):1042–1056, 2007.

[84] H. Sachs, D. M. Cvetkovic, and M. Doob. *Spectra of graphs*. Academic Press, 1980.

[85] M. Saerens, F. Fouss, L. Yen, and P. Dupont. The principal components analysis of a graph and its relationships to spectral clustering. *In Proceedings of the European Conference on Machine Learning*, pages 371–383, 2004.

[86] R. Salakhutdinov and I. Murray. On the quantitative analysis of deep belief networks. *In Proceedings of the International Conference on Machine Learning*, pages 872–879, 2008.

[87] G. Schwarz. Estimating the dimension of a model. *Annals of Statistics*, 6:461–464, 1978.

[88] G. L. Scott and H. C. Longuet-higgins. Feature grouping by relocalisation of eigenvectors of the proximity matrix. *In Proceedings of the British Machine Vision Conference*, pages 103–108, 1990.

[89] G. L. Scott and H. C. Longuet-Higgins. An algorithm for associating the features of two images. *In Proceedings of the Royal Society of London Series B*, 224(1309):21–26, 1991.

[90] C. E. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27(3):379–423, 1948.

[91] L. S. Shapiro and J. M. Brady. Feature-based correspondence: an eigenvector approach. *Image and Vision Computing*, 10:283–288, 1992.

[92] J. Shi and J. Malik. Normalized cuts and image segmentation. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 731–737, 1997.

[93] A. Shokoufandeh, S. J. Dickinson, K. Siddiqi, and S. W. Zucker. Indexing using a spectral encoding of topological structure. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 491–497, 1999.

[94] R. Sibson. Locally equiangular triangulations. *Computer Journal*, 21:243–245, 1978.

[95] S. T. Tokdar and R. E. Kass. Importance sampling: a review. *WIREs Computational Statistics*, 2(1):54–60, 2010.

[96] D. A. Tolliver and G. L. Miller. Graph partitioning by spectral rounding: applications in image segmentation and clustering. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1053–1060, 2006.

[97] A. Torsello. An importance sampling approach to learning structural representations of shape. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–7, 2008.

[98] A. Torsello and D. L. Dowe. Learning a generative model for structural representations. *In Proceedings of the Australasian Joint Conference on Artificial Intelligence*, pages 573–583, 2008.

[99] A. Torsello and E. R. Hancock. Learning shape-classes using a mixture of tree-unions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(6):954–967, 2006.

[100] G. D. Tourassi, E. D. Frederick, M. K. Markey, and Floyd C. E. Application of the mutual information criterion for feature selection in computer-aided diagnosis. *Medical Physics*, 28:2394–2402, 2001.

[101] S. Umeyama. An eigendecomposition approach to weighted graph matching problems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(5):695–703, 1988.

[102] J. von Neumann. *Mathematical foundations of quantum mechanics*. Princeton University Press, 1955.

[103] J. D. Wang, Y. Q. Jia, X. S. Hua, C. S. Zhang, and L. Quan. Normalized tree partitioning for image segmentation. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2008.

[104] D. White and R. C. Wilson. Spectral generative models for graphs. *In Proceedings of the International Conference on Image Analysis and Processing*, pages 35–42, 2007.

[105] D. White and R. C. Wilson. Parts based generative models for graphs. *In Proceedings of the International Conference on Pattern Recognition*, pages 1–4, 2008.

[106] R. C. Wilson and E. R. Hancock. Structural matching by discrete relaxation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(6):634–648, 1997.

[107] R. C. Wilson, E. R. Hancock, and B. Luo. Pattern vectors from algebraic graph theory. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(7):1112–1124, 2005.

[108] R. C. Wilson and Ping Zhu. A study of graph spectra for comparing graphs and trees. *Pattern Recognition*, 41:2833–2841, 2008.

[109] A. K. C. Wong, J. Constant, and M. L. You. Random graphs. *In Proceeds of the Syntactic and Structural Pattern Recognition: Theory and Applications*, pages 179–195, 1990.

[110] B. Xiao and E. R. Hancock. A spectral generative model for graph structure. *In Proceedings of the Joint IAPR International Workshops on Structural and Syntactic Pattern Recognition and Statistical Techniques in Pattern Recognition*, pages 173–181, 2006.

[111] B. Xiao, E. R. Hancock, and R. C. Wilson. Graph characteristic from the heat kernel trace. *Pattern Recognition*, 42:2589–2606, 2009.

[112] B. Xiao, R. C. Wilson, and E. R. Hancock. Object recognition using graph spectral invariants. *In Proceedings of the International Conference on Pattern Recognition*, pages 1–4, 2008.

[113] H. Yamaguchi and K. Tanaka-Ishii. A minimum description length approach to statistical shape modeling. *IEEE Transactions on Medical Imaging*, 21(5):525–537, 2002.

[114] H. Yamaguchi and K. Tanaka-Ishii. Text segmentation by language using minimum description length. *In Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 969–978, 2012.