Parts Based Generative Models for Graphs

David White and Richard C. Wilson
Department of Computer Science
University of York
Heslington, York, UK
wilson@cs.york.ac.uk

Abstract

Generative models are well known in the domain of statistical pattern recognition. Typically, they describe the probability distribution of patterns in a vector space. In contrast, very little work has been done with generative models of graphs because graphs do not have a straightforward vectorial representation.

In this paper we examine the problem of creating generative distributions over sets of graphs. We model the variation in a set of graphs by observing which subgraphs are present in each graph and how these subgraphs are connected. By performing clustering on the subgraphs we can group those with similar structure. Distributions are then defined on the clusters present in each graph, which subgraphs are present in each cluster and the way subgraphs are connected. New graphs can then be generated by sampling from the distributions. We show the utility of our approach on synthetically generated point sets and point sets derived from real-world imagery of articulated objects.

1 Introduction

Generative models are well known in the domain of statistical pattern recognition. Typically, they describe the probability distribution of patterns in a vector space. The individual patterns are defined by vectors and so the resulting features of the pattern are well defined. Graphs are not naturally represented in a vector space since there is no natural labelling of the vertices of the graphs - different labellings lead to different representations of the graph structure.

This problem can be overcome by computing the correspondences between graphs in a set. This process allows graph similarity[7] to be assessed. For example, Gold and Rangarajan[2] describe a method of graph matching by enforcing two-way assignment constraints.

The problem of describing a set of graphs via a representative has been addressed by Jiang et al[3, 4], who define the median of a set of graphs to be the graph which has the minimum sum-of-distances to all the graphs in the set. This definition is equivalent in a sense to the mean of a set of points. Ferrer et al[1] describe a spectral method to compute the median graph.

There are a number of methods in the literature for representing graphs in a vector space. Kosinov and Caelli[5] have used properties of the spectral decomposition to represent graphs and Shokoufandeh et al[9] has used eigenvalues of shock graphs to index shapes. Wilson and Hancock have shown[11] how permutation invariant polynomials can be used to derive features which describe graphs and make full use of the available spectral information. However, it is difficult to reconstruct graphs from these representations.

Luo, Wilson and Hancock[6] directly exploit the adjacency matrix by converting it into a long-vector. An initial correspondence step is used to align the adjacency matrix and the long-vectors are analysed using the eigenmodes. Xiao and Hancock[12] have used the eigenvalues and eigenvectors of the heat kernel to construct the requisite vectors before constructing a Normal distribution in the vector space. While both of these methods construct generative models over the vector space, neither is able to produce reconstructed graphs satisfactorily.

White and Wilson [10] show how to construct a generative model by defining distributions on the vector spaces of the eigenvalues and eigenvectors of graphs. By creating separate distributions for the eigenvalues and eigenvectors, graphs can be generated that are similar to those in the sample set but also novel in structure.

In this paper we describe a parts based approach to generating graphs. We model the variation in a set of graphs by observing which subgraphs are present in each graph and how these subgraphs are connected. To generate new graphs we can then sample from the distributions defined on the subgraphs and connections between them. This approach lends itself to graph types that are easily decomposed, for example, graphs of chemical structures or point sets of objects. The algorithm excels on graphs with small groups of densely connected vertices where the small groups are sparsely connected. By decomposing the graphs we gain an accurate model for these types of data.

2 Approach

We describe our algorithm in two sections. First we discuss how the sample graphs are prepared for use with the model, then we describe the models themselves and how we can sample from them to generate a new graph.

Preparing the Data for use with the Models

The algorithm commences from a sample set of non-directed graphs G which may or may not be weighted. We compute the adjacency matrix representation $\mathbf{A}_k$ for each graph G_k in the sample set.

We seek a partitioning of each sample graph such that each sub-structure i present in the graph is represented by a subgraph adjacency matrix $\mathbf{S}_{ki}$. Each subgraph adjacency matrix represents the subgraph vertices in the partition and the edges wholly within the partition. Clearly this process fails to capture information about the way sub-structures are connected to each other, so we store cut connections between each pair of sub-structures (or more precisely subgraphs) $\mathbf{S}_{ki}$ and $\mathbf{S}_{kj}$ in a *connection matrix* $\mathbf{C}_{kij}$.

This partitioning process is quite flexible and adaptable to the specific data set the process is employed on. For example, it can be done using conventional methods such as Shi & Malik's normalized cut [8] or a more application specific method. The number of partitions can be chosen by visual inspection or by imposing a threshold on the partitioning process.

To simplify working with these different sized subgraphs and connection matrices we find the size of the largest subgraph (n by n) and then pad all other subgraphs $\mathbf{S}_{ki}$ and connection matrices $\mathbf{C}_{kij}$ with zeros so they are n by n in size. The adjacency matrices $\mathbf{A}_k$ are also padded.

Next we cluster the subgraphs such that subgraphs belonging to the same cluster represent similar structure. This information is used to organize the input of subgraphs and connection matrices to the model. Furthermore, it allows us to view the connections between subgraphs as connections between clusters instead.

To cluster the subgraphs we compute the distance between every pair of subgraphs $\mathbf{S}_{ki}$ and $\mathbf{S}_{kj}$ using the weighted graph matching approach of Gold & Rangarajan [2]. These distances are stored in $D(\mathbf{S}_{ki}, \mathbf{S}_{kj})$ and the permutation matrices for matching two subgraphs are recorded in $\mathbf{M}(\mathbf{S}_{ki}, \mathbf{S}_{kj})$. The distance matrix $\mathbf{D}$ is transformed into an affinity matrix and Normalized Cut clustering is performed on it. This results in each subgraph being assigned to a cluster.

The number of cuts performed in the clustering step corresponds to the number of different structures in the sample graphs that are needed for the generative model to accurately convey the sample graph's structures.

With the clusters of subgraphs to hand we compute a reference subgraph ω_c for each cluster c that is chosen to be the one with the smallest distance to all other subgraphs in cluster c. We can then use ω_c and the match matrices $\mathbf{M}$ to align each subgraph to the reference subgraph for that subgraph's cluster. Since the vertices of each subgraph have been permuted the connection matrices must also be aligned. The fully aligned and padded sample graphs are denoted $\hat{\mathbf{A}}_k$.

Constructing the Models and Generating new Graphs

In this section we describe how the models are defined for: the clusters represented in a sample graph, the distribution of subgraphs in a cluster and the connections between clusters. Due to the clustering performed on the subgraphs we can now speak about the connections between subgraphs (the connection matrices) in terms of connections between clusters instead.

To create the model of how subgraphs are connected, the connection matrices of each $\hat{\mathbf{A}}_k$ are stacked into a long vector $\mathbf{r}_k$. The placement of each connection matrix in $\mathbf{r}_k$ is very important to ensure that the same cluster to cluster connection matrices are always placed in the same location in each $\mathbf{r}_k$. Therefore each $\mathbf{r}_k$ stores connections between all possible clusters even though some of these connections may not exist for a particular sample graph.

If a sample graph should contain two or more subgraphs from the same cluster then this is accommodated by allowing as many locations in $\mathbf{r}_k$ as are necessary to store all the connection matrices associated with these two (or more) subgraphs.

We then compute the mean $\bar{\mathbf{r}}$ and covariance $\mathbf{\Sigma}_\mathbf{r}$ of the long vectors $\mathbf{R}$. With the covariance matrix to hand we perform an eigendecomposition resulting in two matrices, $\mathbf{\Phi}_\mathbf{r}$ which contains the eigenvectors and $\mathbf{\Lambda}_\mathbf{r}$ which contains the eigenvalues. We now have the information needed about the distribution of the connections between clusters in the sample set.

To model the distribution of the subgraphs (i.e. which sets of clusters are contained in each sample graph) we use a Gibbs Sampler. To prepare the Gibbs Sampler for use we must setup a number of binary states that indicate which sample graph contains subgraphs from which cluster (let us term this a *cluster configuration*). Again if two or more subgraphs from the same cluster are present in a sample graph then this is recorded in the cluster configuration and taken into account when generating a new configuration.

The distribution governing the subgraphs present in a cluster is simple to define. We use a frequency selection method. For example if there is one cluster that contains

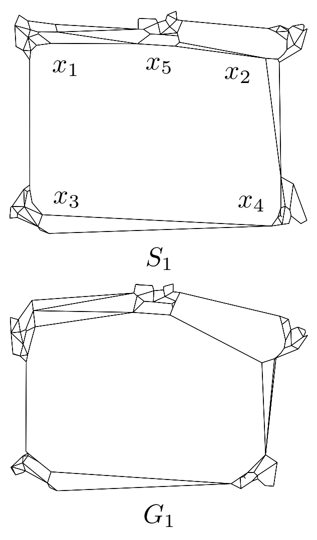

Figure 1. Graphs from the synthetic data set.

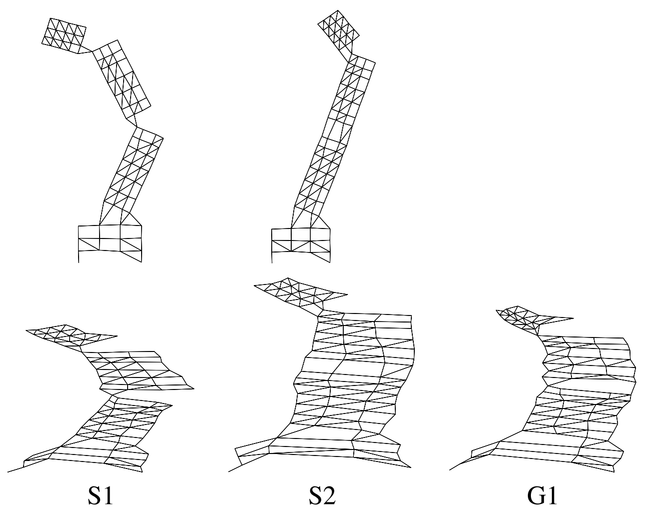

Figure 2. Drawings of the graphs in the articulated object data set .

3 identical subgraphs then that subgraph is 3 times more likely to be selected for that cluster in the new graph than a subgraph that only appears once.

With the three distributions to hand we may now sample from them to generate a new graph. First we must find a cluster configuration using the Gibbs Sampler. For each cluster in the cluster configuration we choose a subgraph. Lastly, we generate the connection matrices.

To generate the connection matrices we generate a new set of connections between all cluster pairs then select the connections that are required for our new graph. The generated set of connections between all cluster pairs is denoted $\mathbf{r}_g$ and computed using the distribution of vectors in $\mathbf{R}$, $\mathbf{r}_g = \bar{\mathbf{r}} + \mathbf{\Phi_r b}$. The parameter vector $\mathbf{b}$ is created by sampling from the normal distribution with zero mean and variance given by the diagonal elements in $\mathbf{\Lambda_r}$. If a connection matrix between a particular pair of clusters is required then it is retrieved and reshaped from the appropriate location in $\mathbf{r}_g$.

Due to the continuous nature of the normal distribution some noise in the connections might have been introduced in generating $\mathbf{r}_g$. To resolve this we use thresholding to remove connections with very small weights. Furthermore, if the sample graphs contained discrete connections then thresholding will be required to recover discrete edges.

3 Results

We show the utility of our approach by applying it to graphs constructed from point sets. Two data sets are used in this section: the first set is constructed from synthetically generated point sets while the second set uses points extracted from real-world images. We use point sets since it provides a way to easily visualize both sample and generated graphs using either an MDS or optimization based drawing technique.

Synthetic Data

Our synthetic data is constructed from 5 different small point sets $x_1, ..., x_5$ each containing 15 points. The point sets $x_1, ..., x_5$ are arranged in a 2D space to make the full point set X_k. A small amount of noise is added to the points in each X_k. Each point set x_i will be represented by a cluster in the algorithm and the way they are arranged will be represented by the connection matrices. We compute the Gabriel graph S_k for each point set X_k from which the algorithm commences. The weights on the edges of the Gabriel graph are set to the distance between the two vertices they connect. The partitioning of each graph is accomplished using Normalized Cut. MDS drawings of two of the four sample graphs (S_1 and S_2) are shown in figure 1 along with the positions of the small point sets.

Figure 1 also shows generated graphs G_1 and G_2. The cluster configurations for these two graphs are the same as the cluster configuration in S_1 and S_2. Notice that the connections in G_1 on the left of the subgraph indicated by x_5 are identical to those in S_1 and the connections on the right are identical to those in S_2. Also observe that the connections between x_2 and x_4 are different to those in any S_k and are in fact a combination of the connections between x_2 and x_4 in S_1 and S_2. This is an example of how the connections between subgraphs can be statistically mixed. Graph G_2 has connections identical to S_2 but different subgraphs have been chosen to represent some clusters. This is clearly seen in the subgraph near x_1.

Real-world Data

To show the utility of our approach on real world data we use images of articulated objects. These objects have multiple joints about which their various parts can move and are therefore an ideal choice for our approach. Our approach will partition the full object into its various parts and then model the way the parts are connected.

We acquire point sets from the real world images through a motion capture preprocessing step. Due to the more complex graph structures present in this data we must employ an optimization based drawing method.

The data set for this experiment consists of 18 images of an object in different articulations. The Gabriel graphs constructed for two of these images are shown on the top row of figure 2. The bottom row shows the optimization based drawing of the graph above it, S1 and S2. Also shown in this figure is a generated graph, G1. Figure 3 shows a PCA projection of the connection matrices for each graph in the sample set and 100 generated graphs. Each sample graph is depicted by its Gabriel graph and the generated graphs are marked with crosses. The generated graphs span the whole space of the sample graphs. The graphs shown in figure 2 are labeled correspondingly in the PCA plot. Notice the locations of S1, S2 and G1 in the PCA plot and the similarities of their structure when drawn in figure 2. On the plot, similar graphs are grouped quite tightly as is the case of the graphs near S2. The variation cause by the bottom articulation of the object does not affect the graph as strongly as the upper two articulations. This is why the graphs grouped around S2 have varying bottom articulations but the top two articulations are the same. Similar trends can be observed for the other groups of graphs in the sample set.

4 Conclusions

In this paper we have shown how to construct a parts based model of a set of graphs that allows new graphs to be generated. This is accomplished by subdividing each graph into a number of subgraphs and then basing the model on these subgraphs and the connections between them. The approach is effective on data types that lend themselves to a partitioning process, namely chemical structures, point sets of articulated objects and point sets of scenes. New graphs are generated by sampling from three different distributions; firstly to find the configuration of clusters that will be present in the new graph, secondly to choose which subgraphs will represent each cluster and finally to generate the connections between the clusters chosen. We show the utility of our approach on synthetically generated data and real-world data. Both data sets take the form of point sets since this provides a way to easily visualize the generated graphs.

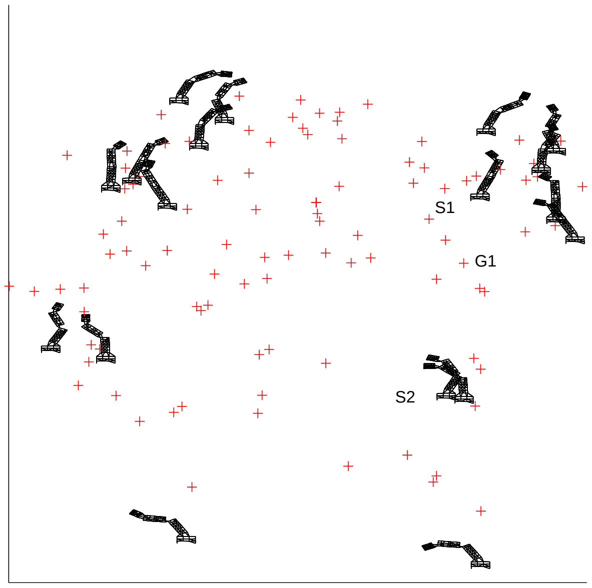

Figure 3. A PCA projection of graphs in the articulated object data set.

References

[1] M. Ferrer, F. Serratosa, and A. Sanfeliu. Synthesis of median spectral graph. In *IbPRIA*, pages 139–146, 2005.

[2] S. Gold and A. Rangarajan. A graduated assignment algorithm for graph matching. *PAMI*, 18:377–388, 1996.

[3] X. Jiang, A. Münger, and H. Bunke. Computing the generalized median of a set of graphs. In *Proc. 2nd IAPR workshop on Graph Based representations*, pages 115–124, 1999.

[4] X. Jiang, A. Münger, and H. Bunke. On median graphs: Properties, algorithms and applications. *PAMI*, 23(10):1144–1151, 2001.

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

[6] B. Luo, R. C. Wilson, and E. R. Hancock. A linear generative model for graph structure. In *Graph-based Representations in Pattern Recognition*, 2005.

[7] A. Sanfeliu and K. S. Fu. A distance measure between attributed relational graphs for pattern-recognition. *IEEE Transactions on Systems, Man and Cybernetics*, 13(3):353–362, 1983.

[8] J. Shi and J. Malik. Normalized cuts and image segmentation. *CVPR*, pages 731–737, 1997.

[9] A. Shokoufandeh, S. Dickinson, K. Siddiqi, and S. Zucker. Indexing using a spectral coding of topological structure. In *CVPR*, number 491–497, 1999.

[10] D. White and R. C. Wilson. Spectral generative models for graphs. *ICIAP*, 0:35–42, 2007.

[11] R. C. Wilson, E. R. Hancock, and B. Luo. Pattern vectors from algebraic graph theory. *PAMI*, 27(7):1112–1125, 2005.

[12] B. Xiao and E. R. Hancock. A spectral generative model for graph structure. In *SSSPR*, 2006.