Data Clustering with a Relational Push-Pull Model

Adam Anthony
University of Maryland Baltimore County
Department of Computer Science and Electrical Engineering
1000 Hilltop Circle
Baltimore, MD 21250
aanthon2@umbc.edu

Marie desJardins
University of Maryland Baltimore County
Department of Computer Science and Electrical Engineering
1000 Hilltop Circle
Baltimore, MD 21250
mariedj@cs.umbc.edu

ABSTRACT

Relational data clustering is the task of grouping data objects together when both features and relations between objects are present. We present a new generative model for relational data in which relations between objects can have either a binding or separating effect. For example, with a group of students separated into gender clusters, a "dating" relation would appear most frequently between the clusters, but a "roommate" relation would appear more often within clusters. In visualizing these relations, one can imagine that the "dating" relation effectively *pushes* clusters apart, while the "roommate" relation *pulls* clusters into tighter formations. We use simulated annealing to search for optimal values of the unknown model parameters, where the objective function is a Bayesian score derived from the generative model. Experiments on synthetic data demonstrate the robustness of the model.

Categories and Subject Descriptors

I.5.3 [**Pattern Recognition**]: Clustering—*Algorithms*

Keywords

Relational Data Clustering, Generative Model, Simulated Annealing

1. INTRODUCTION

Relational data clustering is a form of relational learning that clusters data using the relational structure of data sets to guide the clustering. Many approaches for relational clustering have been proposed recently with varying results.The common assumption in most of this research is that *relations have a binding tendency.* That is, edges are assumed to appear more frequently within clusters than between clusters.

This binding quality may be too strong an assumption. Bhattacharya and Getoor (2005) acknowledge that it is possible for a relation to provide "negative evidence," where the presence of a relation between two objects implies that the objects belong in different clusters. If most of the edges in a relational set provide negative evidence, then this set should be considered to have a *separating*, rather than a binding, tendency. Consider a social network of university students containing both men and women. If the social network is partitioned by gender, edges for a dating relation will appear most frequently between the clusters. This can be visualized as an approximately bipartite structure between the two clusters that "pushes" them apart. Edges for a roommate relation will appear most frequently within the clusters. This binding relation can be visualized as a net that "pulls" the objects in a cluster closer together.

Given that a relation can have either of these two tendencies, accurate clustering in relational data requires determining whether each relation tends to bind or separate objects. Bhattacharya and Getoor (2005) and others have used domain-specific knowledge to identify and process negative evidence. The contribution of our research is a domain-independent model that handles the case where the tendency of each relation uses a generic form of background knowledge about the types of relations in a domain to guide the clustering.

2. TERMINOLOGY

Objects in our model exist in two spaces: the *feature space* and the *relation space.* The object's position in the feature space is specified by its feature vector. The object's position in the relation space is specified by its relations to other objects. We represent an input data set D for our model in two parts: $D = \langle O, R \rangle$. O is a set of objects where each object $o \in O$ is represented as a pair that is composed of a unique identifier and a feature vector. R is a set of edge sets, where each edge set corresponds to a different kind of relation. Each edge set uses the identifiers from the feature space to represent relations between objects as identifier-identifier pairs. Currently, the model supports undirected, unweighted edges, with support for directionality and edge weights to be added in future work. Occasionally, we will use a *relational graph*, $G_t = \langle O, R_t \rangle$, which is a graph containing all data objects and an edge set $R_t \in R$. This is a convenient way to refer to all edges of a particular kind of relation.

3. THE RELATIONAL PUSH-PULL MODEL

We propose the Relational Push-Pull Model (RPPM), a Bayesian model that can be used to find clusterings in relational data. Figure 1 is a graphical representation of the model that shows the dependencies for a pair of objects, l

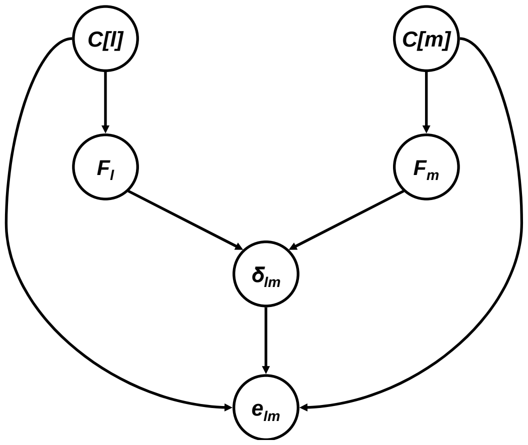

Figure 1: Graphical model of the dependencies in the Relational Push-Pull Model

and m. Object features depend on the cluster the object belongs to, modeled as a mixture of Gaussians (Russell & Norvig, 2003). To formulate an edge distribution, we use the concept of *existence uncertainty* (Getoor et al., 2002) for relations. For each pair of objects l and m in a data set, we calculate the probability that a link exists between the objects. Edge existence is represented by the node labeled e_{lm}.

Ultimately, we want to determine the probability of a clustering, given the objects in the feature space and the edges in the relation space, i.e. $Pr(C \mid O, R)$. We will use this value as an objective function for a local search method. Using Bayes' rule, we can compute this probability as:

$$Pr(C \mid O, R) = \alpha \, Pr(R \mid C, O) Pr(C \mid O), \qquad (1)$$

where α is a normalizing constant. In the following sections, we will first explain our edge existence model in detail and provide a way to calculate first term, $Pr(R \mid C, O)$. Then, we will discuss our feature space model, and how to use it to compute the second term, $Pr(C \mid O)$.

3.1 Modeling Relation Graphs as Push or Pull

The edge existence probability model that we use is slightly different from that of Getoor et al. (2002). In their work, edge existence depends only on the class labels of the objects the edge would connect. Our notion of edge existence also incorporates the concept of *relational autocorrelation* (Jensen & Neville, 2002), the tendency for feature values in linked objects to have similar values. In Figure 1, we represent autocorrelation by placing arcs from each object to a node representing the distance δ_{lm} between them, and then an arc from the distance node to the edge existence node. To motivate the edge existence model, we first discuss the benefits of class-based edge existence and relational autocorrelation separately, then show how the two ideas can be combined.

One way to model edge existence using only cluster labels is to specify two constant probabilities, IC and OC. IC (*in cluster*) is the probability of an edge existing between l and m if the objects are in the same cluster, and OC (*out cluster*) is the probability that the edge exists if they are in different clusters. Figure 2 shows the conceptual relationship between the two values. Pull-type relations are created when the value of IC is much greater than OC. Conversely, push-type relation graphs result when the value of IC is much smaller than OC. The gray region illustrates that when IC is very similar to OC, the relation graph's tendency is ambiguous. The advantage of our model is that it can be used to quantitatively represent both push and pull relations.

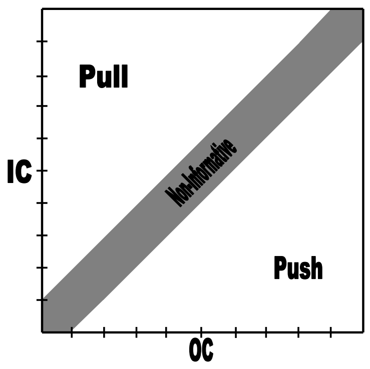

Figure 2: The relationship between IC and OC. When one is high, and the other is low, we can infer a pull-type relational graph or a push-type relational graph. The shaded region indicates the area where it is difficult to infer the graph type (non-informative type).

Jensen and Neville (2002) claim that relational autocorrelation is a common phenomenon in relational data sets. A simple approach to exploiting autocorrelation for edge existence is to use the distance between two objects. For example, it is intuitive to represent the edge existence probability as a decreasing function of distance, such as $Pr(e_{lm}|\delta_{lm}) = e^{-\delta_{lm}}$. This models the situation where objects similar in the feature space are more likely to be related. For example, people who are closer in age are more likely to be roommates. This method for edge existence has the advantage of emphasizing the influence of autocorrelation, but it does not model the relation graph's tendency to push or pull.

We can get the benefits of both approaches if we make edge existence dependent on both cluster membership and distance. We model this dependence with the following formula:

$$Pr(e_{lm} \mid l \in C_i, m \in C_j, \delta_{lm}, \lambda_{I_t}, \lambda_{O_t}) = \begin{cases} IC(\delta_{lm}) = e^{-\delta_{lm}} - (1 - \lambda_{I_t}) & \text{if } C_i = C_j; \\ OC(\delta_{lm}) = e^{-\delta_{lm}} - (1 - \lambda_{O_t}) & \text{if } C_i \neq C_j. \end{cases} \qquad (2)$$

Instead of being constant, IC and OC are functions that

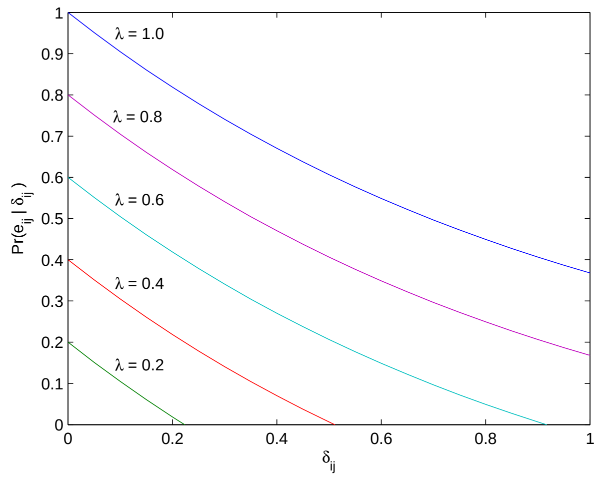

Figure 3: Probability plots of edge existence, for various values of λ

are selected depending on the cluster membership of l and m. λ_{I_t} and λ_{O_t} are parameters that determine which of the two functions is higher, and thus whether IC relations or OC relations are more likely. Figure 3 shows a plot of $e^{-\delta_{lm}}$ for various values of λ. We use the expression $(1-\lambda)$ above to make the value of each lambda more intuitive for the reader: a higher lambda corresponds to a higher probability of edge existence. We also enforce a cutoff when the function becomes less than zero, so that all values returned are positive. Equation 2 assumes that each cluster has the same IC function, and that edge generation between any two clusters has the same OC function. It would be simple to extend Equation 2 to specify functions for all possible combinations of cluster membership, if necessary.

We can compute the probability distribution of a relational graph for one edge type (G_t) as a product of each edge's probability of existence:

$$Pr(G_t \mid C, O) = \prod_{e_{l,m} \in G_t} Pr(e_{lm} \mid \ldots) \times \prod_{e_{l,m} \in O \times O, \notin G_t} \{1 - Pr(e_{lm} \mid \ldots)\} \tag{3}$$

Where $Pr(e_{lm} \mid \ldots)$ is calculated as in Equation 2. The conditioning contexts are omitted to improve readability.

Finally, the probability distribution of the entire relation space can be calculated as the product of the distribution of each relation graph with type t:

$$Pr(R \mid C, O) = \prod_{\text{all t}} Pr(G_t \mid C, O), \tag{4}$$

which completes the derivation of the first term in Equation 1.

3.2 Modeling Features

As stated previously, the feature space is modeled as a mixture of Gaussians. For each cluster k, we define a weight w_k, a center mean μ_k and covariance matrix Σ_k. Under this model, if an object is assigned to a cluster, it is assumed that its features are sampled from the distribution specified by that cluster's parameters. Therefore, we can model each object o's existence in the feature space as the probability in its cluster's Gaussian:

$$Pr(o \mid o \in C_k) = \frac{1}{(2\pi)^{d/2} |\Sigma_k|^{1/2}} e^{\frac{1}{2}(o-\mu_k)^T \Sigma_k^{-1} (o-\mu_k)} \ . \tag{5}$$

Given a fixed assignment of objects to clusters, and the set of weights $\mathbf{w} = \{w_1 \ldots w_k\}$ for each cluster, we can specify the probability of C using a multinomial distribution:

$$Pr(C \mid N, \mathbf{w}) = \left(c_1 \overset{N}{\cdots} c_k \right) w_1^{c_1} \cdots w_k^{c_k}, \tag{6}$$

where N is the total number of objects, and c_k is the number of objects assigned to cluster k.

From Equations 5 and 6, we can derive the second term of Equation 1, $Pr(C \mid O)$ as:

$$\begin{aligned} Pr(C \mid O) &= \alpha \, Pr(O \mid C) \times Pr(C) \\ &= \alpha \prod_{o \in O} Pr(o \mid o \in C_k) \times Pr(C \mid N, \mathbf{w}) \ . \end{aligned} \tag{7}$$

3.3 Learning an RPPM Model

The RPPM defines a large model with cyclic dependencies and several parameters. Additionally, since we cannot assume object independence, learning this model can be difficult. The major challenge comes from the pairwise dependence for edge existence between all pairs. For example, with k clusters and n objects, a naïve EM algorithm would require $O(kn^2)$ calculations in the E-step to accommodate all of the potential relations in the model. In future work, we intend to further investigate faster methods for EM approximation.

As a first approach, we decided to use simulated annealing to evaluate our model's potential. We chose simulated annealing over a more basic hill-climbing approach because of the tradeoff between searching in the feature space and searching in the relation space. It is possible that a maximization in one space will reduce the term in Equation 1 for the other space. This results in many local maxima, making simulated annealing an appropriate choice. In Section 4, we show empirically that simulated annealing can find a global maximum that represents the correct clustering.

We currently assume fixed values of λ_{I_t} and λ_{O_t} for each relation. In our experiments, we sometimes provide the correct values for λ_{I_t} and λ_{O_t}, incorrect values for λ_{I_t} and λ_{O_t}, or use an estimation procedure, which we describe later. We also assume that each edge is labeled with its relation type and that the number of clusters (k) is known and fixed. The simulated annealing algorithm, then, must search for an optimal clustering (assignment of each object to a cluster) and the parameters (w_k, μ_k, Σ_k) for each cluster, using Equation 1 as an objective function.

Our specific implementation of simulated annealing uses a linear cooling schedule $T = 1 - \frac{t}{R}$, where t is the elapsed time and R is a sufficiently large value (set to 10,000 in our implementation) that controls the rate at which the temperature decreases. At each time step t, the simulated annealing algorithm generates a successor in the following way.

1. Select $k \times \tau$ objects uniformly from the feature space as candidates for moving. τ is an integer ≥ 1 that controls the number of objects selected. Intuitively, this step selects, on average, τ objects from each cluster. We currently set τ to be 4 when T is high, and 1 when $T < 0.25$.

2. For each selected candidate, choose a cluster to assign it to randomly with uniform probability—it may be assigned to the same cluster it is currently assigned to,

3. For each cluster k, compute the maximum likelihood estimates for w_k, μ_k and Σ_k.

If the successor improves the objective function, it is always chosen. If not, we calculate the loss L as the difference between the current and generated states. Then we accept this lower-valued state with probability $e^{L/T}$.

Since the simulated annealing algorithm does not search for the λ parameters, we introduce the concept of a *relation type hypothesis*. Assume that a user's background knowledge includes an intuition about a relation type's push or pull tendency, but not the proper values of λ_I and λ_O for each relation type. To form a hypothesis, first estimate the edge density d of the graph directly, or by sampling with replacement. This narrows the possible values we can select for each λ_I and λ_O, given an initial intuition. In the coordinate plane in Figure 2, the line $IC = -OC + d$ shows all possible values of IC and OC that produce a density d. Using the initial intuition, choose a ratio of IC to OC that is appropriate. For example, $r = IC/OC = 0.2$ is a good guess for a strong pull relation. Solving the linear equation using r and d is a good heuristic for guessing the values of λ_I and λ_O. Substituting $OC = r * IC$, we get $IC = d/(1 + r)$ and $OC = r * IC = rd/(1 + r)$. For example, assume that we hypothesize a push/pull ratio of 0.2 for the strong-pull relation, and the density of this relation is 0.15. Using these formulas, we can now guess that $\lambda_I = 0.125$ and $\lambda_O = 0.025$.

4. EXPERIMENTAL RESULTS

Our experiments demonstrate the use of a relational hypothesis to cluster data in domains where the tendency of a relation is clear, such as a citation link in a bibliographic database. It is generally assumed that papers of the same topic are more likely to have a citation between them than papers of different topics. In such a situation, we may state, for example, that $\lambda_{I_{\text{cite}}} = 0.75$ and $\lambda_{O_{\text{cite}}} = 0.15$ to indicate a pull-type relation in RPPM. Another possible use of a relation hypothesis is to reveal alternative clusterings. If we first run the algorithm assuming a relation is a pull-type, we will discover an optimal clustering, given that hypothesis. If we then run the algorithm assuming the same relation is a push-type, we will find a different clustering. A quantitative and/or qualitative comparison of the two solutions could then be performed to decide which hypothesis is more useful.

To evaluate all experiments, we use a simple accuracy calculation computed as the average number of objects clustered correctly. Because we are using artificial data, we pre-label the data as it is generated. To calculate the accuracy, we count the objects in each cluster that have have the same label as the cluster number. For example, we count all the objects labeled '0' that are in cluster zero. We then divide the sum of these counts by the total number of objects. Because we do not know which cluster best represents cluster zero, one, etc., we perform the accuracy calculation for all permutations of clusters and return the maximum score. We also acknowledge that this method will not work for real-world data sets. In the future, we intend to implement methods such as the Adjusted Rand Index to evaluate clusterings.

Our initial experiments use artificial data for evaluation that is generated according to the RPPM specification:

```
input parameters $\{\mathbf{w}, \mu, \mathbf{\Sigma}, \lambda_{\mathbf{IC}}, \lambda_{\mathbf{OC}}\}$
for each object $o$:
  1. Choose a cluster membership $o \in C_k$,
     where $k$ is chosen with $Pr(o \in C_k) = w_i$
  2. Generate the feature vector for $o$ as a sample
     from the multivariate Gaussian with
     parameters $\mu_k$ and $\Sigma_k$
for all object pairs $(l, m)$:
  for each relation type $t$:
     1. with $Pr(e_{lm} | l \in C_i, m \in C_j, \delta_{lm}, \lambda_{ICt}, \lambda_{OCt})$,
        place an edge $e_{lm}$ in the set $RG_t$
```

The data size varies from 250 to 1500. The Gaussian distributions chosen for generating the data were selected with parameters that would ensure a significant overlapping of the clusters so that it is difficult to cluster using just the feature space portion of the model. The parameters in the relation space are given specific values to simulate the effect of clustering data with different relation types.

4.1 The Impact of an Incorrect Hypothesis

We have already claimed that many researchers make the assumption that all relations have a pull tendency. Our first set of experiments tests the accuracy of clustering under several different modeling assumptions:

1. Ignoring relations,
2. Assuming all relations are push relations,
3. Assuming all relations are pull relations,
4. Using the true λ values for all relations, and

The generated data sets have 250 two-dimensional objects in two clusters with three different relation types. The first relation type is a pull-type relation, with $\lambda_I = 0.8$ and $\lambda_O = 0.2$; the second relation type is a push-type relation, with $\lambda_I = 0.3$ and $\lambda_O = 0.6$; and the third is neither push nor pull, with $\lambda_I = 0.3$ and $\lambda_O = 0.3$. The correct hypothesis assumes the parameters listed above. The all-pull hypothesis assumes the parameters are $\{0.9, 0.1\}$ for all three graphs and the all-push hypothesis assumes the parameters are $\{0.1, 0.9\}$ for all three graphs.

Table 1 shows the clustering accuracy for each of the above cases for three different runs: using the feature data only, using the graph data only, and using both data sources together. Accuracy is calculated as the number of objects clustered correctly divided by the total number of objects, averaged over results for ten unique data sets.

The first observation is that the graph and feature scores have very similar values to the graph-only scores for all three hypotheses. In fact, a T-test shows no significant difference between the graph and feature scores and the graph only scores, with P-values no lower than 0.33. This may be because the hypotheses are very strong assumptions and dominate the graph and feature scores. It may also be the case that the graph score is sufficient for clustering. If this is the case, we could either eliminate the feature score or alter the model to make the contribution of information in the feature space more significant. We intend to perform additional tests to determine why the feature score has such a small impact.

A second observation is that the all-pull hypothesis outperforms the all-push hypothesis for both graph-only and graph and features; T-tests comparing the values in each row (i.e., graph only, all pull compared to graph only all push) all had P-values less than .01. This is most likely due to the higher edge density in the pull graph than the push graph and the fact that our formula for edge existence favors nodes that are in the same cluster because it favors nodes that are closer together. Third, the all-pull accuracy for both graph-only and graph and features outperforms the features-only accuracy, and the all-push accuracy for both graph-only and graph and features is worse than the features-only accuracy. The difference for the all-pull hypothesis is not statistically significant, but the difference for the all-push hypothesis is, with P-values less than 0.02 in both cases. This means that if we assume all graphs are pull graphs in our model, we aren't likely to get worse results than when relations are ignored. But if we assume all-push, we can expect slightly worse results.

4.2 Varying the Strength of the Push-Pull Tendency

This experiment investigates how much an edge set contributes to the final clustering, depending on the strength of its tendency. For this trial, we generated a data set with 1000 four-dimensional objects in four clusters. We ran our algorithm five times with the same feature set, and different edge sets. The sets vary from a strong pull-type relation to a strong push-type relation. Figure 4 summarizes the results. As expected, performance decreases as the strength of the tendency decreases. This initial experiment is intended only to demonstrate the qualities of the model. We intend to run further experiments to cover a broader range of λ_I and λ_O, as well as to complete a larger number of trials so that we can report the statistical significance of these findings.

4.3 Varying the Edge Density

Real-world relational data sets tend to have sparse edge sets. In this trial, we are interested in how sensitive our model is to varying edge density. We use the same feature space as in the previous trial, but now our edge sets vary in density.

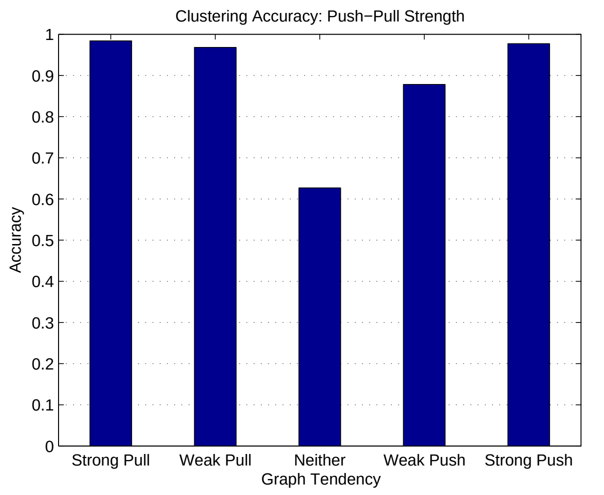

Figure 4: The impact of varying push-pull relation strengths

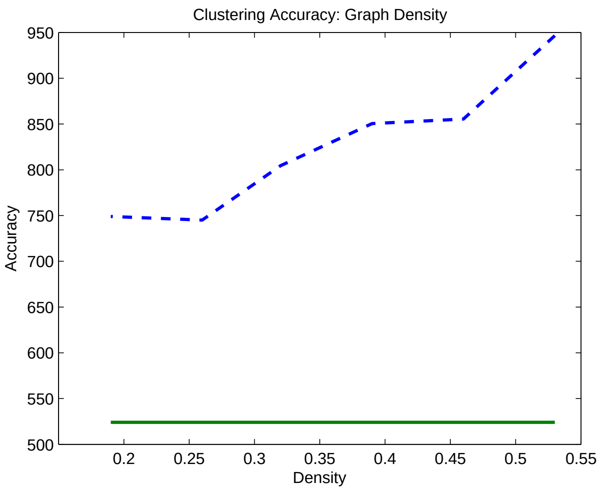

Figure 5: Varying densities results in decreasing performance

We do this by setting a fixed ratio of 0.2 between λ_I and λ_O, then increasing λ_I in increments of 0.1 and increasing λ_O proportionately. Figure 5 shows the expected outcome that performance decreases with density. The features-only line, drawn to show the improvement for each density, is horizontal because we ignore the edges in that case. This initial experiment in varying densities does not cover many values of λ_I and λ_O, nor does it cover many densities. In future work, we would like to explore the impact of relation density further, as well as complete a larger number of trials for each density value.

4.4 Testing the Hypothesis Formation Procedure

In this final test, we simulate a more realistic data set with a larger number of objects and multiple relation types. Here our feature space has 1,500 objects with 10 features. We

Data Source	Correct Hypothesis	All Pull Hypothesis	All Push Hypothesis
Graph and Features	0.9576	0.6640	0.5696
Graph Only	0.9564	0.6676	0.5768
Features Only	0.6432	0.6432	0.6432

Table 1: The impact of a poor hypothesis. When the edge hypothesis is incorrect, the optimal solution is not the correct clustering.

Lambda Values	Accuracy
None (Features only)	0.51
Density-Guess	0.63
Actual Values	0.932

Table 2: Performance of the density-based guess method

have five edge sets with the following push-pull tendencies: strong pull, weak pull, no tendency, weak push, strong push. We do not, however, know these values explicitly. The λ_I and λ_O parameters for each relation type are selected within a fixed range that ensures they have the desired tendency:

- Select λ_I on the range $[0, 1]$,
- Calculate λ_O to have a λ_O/λ_I ratio of [0.25, 0.5, 1.0, 1/0.5, 1/0.25] (plus or minus a noise factor of 0.05 for each term) depending on the desired tendency,
- Generate the data as described at the beginning of Section 4.

Given a data set generated in this way, we can test our hypothesis forming procedure from Section 3.3. For the first graph, we will guess that $r = IC/OC = 0.25$, though it may be as high as 0.3 and as low as 0.2. Then we solve for $IC = d/1.25$ and $OC = .25d/1.25$, where $d = \#\text{edges}/|O|^2$. We complete this process for all five edge types, using the mid-point of the possible values the true lambda parameter could be. Table 2 summarizes the results using the hypothesis formation procedure. The density guess does show a significant improvement over the features-only clustering method, but does not perform well compared to using the actual values. We intend to investigate whether this method can be refined to estimate the λ values with greater accuracy.

5. RELATED WORK

There have been a number of proposed generative models for relational data in recent research. The most similar model to the RPPM is the Latent Group Model (LGM) proposed by Neville and Jensen (2006). LGMs have the same dependency for object features that the RPPM has: each feature value depends solely on the object's cluster membership. Their model differs in their notion of edge existence. They assume that in addition to objects belonging to clusters, objects also belong to a different, hidden grouping in the data that represents the cause of relational autocorrelation. Edges then depend solely on the groups to which its endpoint objects belong. This is similar to our discussion of the constant IC and OC for edge generation, except that in their model the edge exists based on the latent groups instead of cluster membership. The key difference from the RPPM is that LGMs do not consider the distance between two objects when determining edge existence. Their model does have an advantage that it can specify a different probability for edge existence between each specific group, rather than general values for "in-cluster" and "out-cluster" probabilities.

Another generative model is the Probabilistic Relational Model (Getoor et al., 2002). This model extends the naïve Bayes model of independent data objects to include relations by allowing the class label of an object depend on the class label of another object it is linked to. Getoor et al. also introduce the concept of edge existence uncertainty that was adapted for use in this paper. Their model has the advantage that it is flexible in the tasks that it can be used for. For example, the model was originally developed for making predictions of link structure in a data set, but Taskar et al. (2001) use it to cluster relational data. Because the class of one cluster can depend on the class of another, they were able to generate meaningful clusters in a movie database where actors were clustered together in part by the types of movies they acted in, because an actor's class depended on the class of the movies he acted in. Taskar et al.'s method does not incorporate the ideas of edge uncertainty, and the clusterings do not depend on the link structure directly.

Finally, two other models that are less related are those of Kemp et al. (2006) and Kubica et al. (2002). Kemp et al. (2006) proposes the Infinite Relational Model, an extension of stochastic block models to handle the case where the number of clusters is unknown. Their model has the ability to model the probability of an edge existing between or within clusters, but like Neville and Jensen (2006), this value is a constant. The IRM has a unique quality that object features can be represented in the form of a relation so that the process of clustering data objects also results in a clustering of object features. Kubica et al. (2002) present a model where links exist as the result of some event (e.g., a phone call or a meeting). They use this model to detect groups of associated people from a social network. Their model is effective in representing such a situation, but does not generalize well to other problems.

6. FUTURE WORK

We have demonstrated that the Relational Push-Pull Model can be used to perform clustering tasks on certain kinds of relational data. Our experiments show that this is a promising model that is robust to both edge distribution and edge density. Below we describe modifications to the RPPM that could further enhance is utility. We have ordered them in increasing levels of difficulty.

Edge Weight and Edge Directionality: Clustering an undirected, unweighted graph is only our initial approach. Many real-world data sets include directionality and/or weight information for each edge. Examples include food webs, web-page graphs, social networks, and weblog graphs.

Incorporating edge weights involves modifying Equation 2, the edge existence probability. Possible solutions are to use a normalized edge weight as an additional modifier of IC and OC or to specify different equation choices for a specified range of edge weights.

Incorporating directionality is more complex. One solution would be to declare that either direction of an edge is equally probable, and leave the model unmodified. This does not seem like a plausible solution. For example, in food webs, an owl is likely to eat a mouse, but a a mouse is not likely to eat an owl. A second solution is to add a layer of abstraction, leaving the edge existence equation alone and adding a second parameter indicating the likely direction the existing edges point. This approach would mesh nicely with *variable relation tendency strengths*, which we describe next.

Variable Relation Tendency Strengths: Consider a clustering case where there are three clusters. RPPM assumes that the probability of an edge existing inside cluster 1 is the same as an edge existing inside cluster 2 or 3, and that the probability of an edge existing between clusters 2 and 3 is the same as the probability of an edge existing between clusters 1 and 2 or between 1 and 3.

The most intuitive solution to this problem is to specify additional λ_I and λ_O parameters for all the possible pairings, and to also expand Equation 2 to accommodate all pairings. This would add little overhead to the model, and would work well for the relation-type hypothesis clustering method, but, as we discuss next, we would like to be able to search for all parameters in the model, and adding additional parameters would make the search more difficult.

Expectation Maximization Approach: We mentioned earlier that a naïve approach to the Expectation Maximization algorithm would require $O(kn^2)$ calculations at each iteration. This is because for each object, we would have to compute the edge existence probability for that object's relation to all other objects. As a result, the algorithm is infeasible for data sets larger than a few thousand objects. We intend to develop an approximation approach to reduce the number of computations.

Discrete Methods: We also propose the development and analysis of discrete methods for relational data clustering. The RPPM provides a foundation for both theoretical and empirical evaluation of discrete methods we develop.

Heterogeneous Data: The most complex relational data set possible has multiple weighted, directed edges and multiple types of objects, each of which is defined in its own feature space. Feature-based comparison of heterogeneous objects is difficult and sometimes impossible. The RPPM can be used to cluster data without comparing object features, as we showed in our experiments. The results are not quite as good as when features and objects are considered together, but do show some promise for clustering different typed objects together. We intend to explore the performance of this method, as well as ways to extend our model to handle multiple object types explicitly.

7. RESEARCH SCHEDULE

My plan is to extend as much of the future work described above in the next three months. Below, I list a specific schedule for completion of my Master's thesis:

Estimated Completion	Task
March 1 - March 15	• Extend RPPM to support directed, weighted edges. • Process raw food web data, remove or estimate missing feature values
March 15 - April 1	Experimentation on artificial and food we Complete and revise preliminary thesis ch
April 1 - April 15	

8. CONCLUSION

The Relational Push-Pull Model is a unique framework that models the specific tendencies that a relation can have with regards to a specific clustering. This model expands the space of possible clusterings beyond previous works that considered all relations to be pull-type relations. By searching in this expanded space, we showed that better clusterings may be discovered for different data sets. Given additional modifications and further refinement of our search method, we believe that Relational Push-Pull Models will become a useful tool for clustering complex relational data.

9. ACKNOWLEDGEMENTS

This work supported by NSF grant #0545726.

References

Bhattacharya, I., & Getoor, L. (2005). *Entity resolution in graph data* (Technical Report CS-TR-4758). University of Maryland.

Getoor, L., Friedman, N., Koller, D., & Taskar, B. (2002). Learning probabilistic models of link structure. *Journal of Machine Learning Research*, *3*, 679–707.

Jensen, D., & Neville, J. (2002). Linkage and autocorrelation cause feature selection bias in relational learning. *Proceedings of the Nineteenth International Conference on Machine Learning (ICML 2002)* (pp. 259 – 266). Morgan Kaufmann.

Kemp, C., Tenenbaum, J. B., Griffiths, T. L., Yamada, T., & Ueda, N. (2006). Learning systems of concepts with an infinite relational model. *Proceedings of the Twenty-First National Conference on Artificial Intelligence* (pp. 381 – 388). Menlo Park, California: AAAI Press.

Kubica, J., Moore, A., Schneider, J., & Yang, Y. (2002). Stochastic link and group detection. *Proceedings of the*

Eighteenth National Conference on Artificial Intelligence (pp. 798–804). AAAI Press/MIT Press.

Neville, J., & Jensen, D. (2006). Leveraging relational autocorrelation with latent group models. *Proceedings of the Fifth IEEE International Conference on Data Mining*.

Russell, S., & Norvig, P. (2003). *Artificial intelligence: A modern approach*. Upper Saddle River, New Jersey: Prentice Hall. 2 edition.

Taskar, B., Segal, E., & Koller, D. (2001). Probabilistic classification and clustering in relational data. *Proceeding of IJCAI-01, 17th International Joint Conference on Artificial Intelligence* (pp. 870–878). Seattle, US.