Spectral Clustering with Links and Attributes

Jennifer Neville, Micah Adler, David Jensen
Department of Computer Science
University of Massachusetts
Amherst, MA 01003
{jneville|micah|jensen}@cs.umass.edu

Abstract

Relational data offer a wealth of information for identifying groups of similar items. Both attribute information and the structure of relations can be used for clustering. If the data contain *communities*—groups of items that have similar attributes and are also highly inter-connected—a clustering technique that exploits both sources of information simultaneously should produce more meaningful clusters. We investigate this hypothesis in the context of a spectral graph partitioning technique developed for image segmentation. We consider six similarity metrics: one using attributes in isolation, one using only link structure, and four hybrid metrics that combine both sources of information. Through simulation, we find that two of the hybrid metrics achieve superior performance over a wide range of data characteristics. To investigate the mechanisms underlying this achievement, we analyze the spectral decomposition algorithm from a statistical perspective and show that the successful hybrid metrics use the link and attribute information to increase the separation between noisy clusters. We apply the algorithm, using the best hybrid metric, to number of relational datasets and show that resulting clusters exhibit significant *community* structure. Finally, as objective evaluation, we show that the hybrid clusters can be used to significantly improve performance in a related classification task in the genomic domain.

1 Introduction

Clustering is a descriptive task that seeks to identify natural groupings in data. Developing techniques to automatically discover such groupings is an important part of knowledge discovery and data mining research. The majority of data routinely captured by businesses and organizations are relational in nature yet few clustering techniques have been developed to take advantage of both the attribute information and the structure of relationships in the data. Relations between instances indicate an affiliation in the same way that similar attribute values indicate an connection. As such, clustering algorithms that incorporate relational information should be able to produce better groupings than those that examine attribute information in isolation.

Relational data consist of objects (representing people, places, and things) connected by links (representing persistent relationships among objects). For example, a relational

dataset could represent information about the motion picture industry, with objects representing studios, movies, and people (e.g., actors, directors, and producers) and links representing relationships (e.g., actor-in and remake-of). Clustering algorithms are used primarily to describe the data in a set of higher-level patterns (i.e. the clusters themselves). However, clustering algorithms may also be included as a component in a larger knowledge discovery system. In this case, cluster labels may be used to create new attributes for learning predictive models. For example, clustering actors in the movie data may produce groupings that represent abstract types such as *action-hero* or *teenage-idol*. Actor type could then be used to improve predictions of a movie's box office success.

Both data clustering and graph partitioning techniques can be used to cluster relational data—relationships in the data provide graph structure and attribute values on objects provide data information. Conventional data clustering algorithms identify groups of similar items in a dataset based on their attribute values (e.g., [2, 10]). For example, actors can be clustered based on their age, gender and nationality. Traditional graph partitioning algorithms use the structure of a graph to find highly connected components (e.g., [1]). For example, actors may be clustered by the edges that represent starred-with relationships to group actors that star in many movies together. Either approach, used independently, may offer insight into the data but there are also potential benefits to examining attribute and relations at the same time.

This work is focused on finding *communities* in relational data. Community clusters identify groups of items that have similar attributes and are also highly inter-connected. For example in genomic data, a group of genes with similar attributes and many common interactions may all be involved in a similar function in the cell. The underlying assumption is that there is a latent (hidden) cluster variable for each object that influences both the attribute values intrinsic to the object and its relationships to other objects. In particular, objects are more likely to link to other objects in the same cluster than objects in other clusters, and pairs of objects within a cluster are more likely to have similar attribute values than pairs spanning different clusters. An algorithm that examines both link structure and attributes simultaneously should be more robust to noise—combining both sources of information will improve the clustering results.

In this paper, we investigate methods of adapting a spectral graph partitioning technique to incorporate both link structure and attribute information. In particular, we focus on recent work by Shi and Malik on a divisive, hierarchical clustering algorithm that uses spectral partitioning with a *normalized cut* objective function [21]. This technique has been successfully applied in a number of domains, including image segmentation [21] and document clustering [8], and has prompted further investigation into the properties of spectral clustering. Recent findings—facilitated by a long history of work in spectral graph theory (e.g., [4])—include a connection to random walks [17] and preliminary performance bounds [14].

There has been very little work applying spectral techniques to relational domains with a combination of link and attribute information. Existing techniques use either: (1) a *complete* graph where attribute similarity is calculated for all $n \times n$ pairs of objects, or (2) a nearest neighbor graph, where an attribute similarity is calculated for $n \times d$ pairs of objects—each object is connected to a fixed number of other objects determined by spatial locality. Our work differs in that we are specifically trying to include the relational graph structure in the similarity metric. Specifically, we will investigate the design of similarity metrics that can incorporate the two sources of information and explore the characteristics that underlie successful metrics.

The remainder of this paper is organized as follows: First, we provide a statement of the problem, and contrast relational clustering with conventional data clustering and graph partitioning. Next, we describe the spectral clustering algorithm, define six similarity metrics,

and outline the conditions under which algorithm performance will be exact. Then, we analyze performance using synthetic datasets, show that two of the hybrid metrics achieve superior performance, and explore the reasons for these performance gains. Next, we evaluate performance, using the best hybrid metric, on three real-world relational datasets, show that resulting clusters exhibit significant *community* structure, and demonstrate significant performance gains when using the resulting clusters in a related classification task. Finally, we discuss related work in probabilistic modeling and web-page clustering, and conclude.

2 Clustering Relational Data

The goal of this work is to discovering a natural typing over objects (e.g., find groups of *similar* objects) in relational data. This is an unsupervised learning problem where the *correct* grouping is unknown. There are two basic reasons for interest in unsupervised learning problems. The first is that such exploratory data analysis often leads to insight into the nature of the data. The second, is that it may lead to the discovery of features that are useful for future classification tasks.

Conventional clustering algorithms use attribute information to group examples under the assumption that two instances are related if they have similar attribute values. However, relational data have more information available to disambiguate groupings. We hypothesize that links confer a relationship between two instances in the same way that similar attribute values indicate a relationship. As such, clustering algorithms that incorporate link information should be able to produce better groupings. Co-citation analysis [22] is based on a similar hypothesis—if many pages point to a set of pages, then the set pages are likely to address the same topic. Likewise, if a set of pages all point to the same pages, then the set of pages are likely to be semantically related.

Both data clustering and graph partitioning techniques can be used to cluster relational data—relationships in the data provide graph structure and attribute values on objects and links provide data information. Each approach used independently may offer insight into the data but there are also potential benefits to examining attribute and relations at the same time.

First, attributes and structure can be used to cluster for objects playing similar *roles* in the data. Clusters such as these identify groups of items that are similar both in their attributes and their relations to other types of instances. For example, *leading-ladies* have similar gender and salary attributes and also star in many blockbuster movies.

Second, attributes and structure can be used to cluster for *communities* in the data. Community clusters identify groups of items that have similar attributes and are also highly inter-connected. For example in citation data, a group of papers with similar terms and many intra-group citations may indicate an emergent research topic. In genomic data, a group of genes with similar attributes and many common interactions may all be involved in a similar function in the cell.

This work is focused on finding communities in relational data. The underlying assumption is that there is a latent (hidden) cluster variable for each object that influences both the attribute values intrinsic to the object and its relationships to other objects. In particular, objects are more likely to link to other objects within the same cluster than objects in other clusters and objects within a cluster are more likely to have similar attribute values than objects in different clusters.

Given noise-free data generated from the underlying process described above it should be possible to recover the cluster structures using either data clustering or graph partitioning alone. However, noise in either the attribute values or the link structure could reduce the accuracy of clusterings formed from only a single source of information. In the presence of

noisy data, an algorithm that examines both structure and attributes simultaneously should achieve superior results by combining both sources of information.

Both data clustering and graph partitioning techniques can be used to cluster relational data—relationships provide graph structure and attribute values provide data information. Conventional data clustering algorithms identify groups of similar instances in a dataset based on their attribute values (e.g., [2]). Given a dataset of N independent instances, and a set of k attributes, the algorithms assign the objects to a set of clusters such that objects within clusters are *similar* and objects in different clusters are *dissimilar*. There are many different measures of similarity, which are a function of the instances' attribute values—for example, Euclidean distance, cosine similarity, Dice's coefficient, or Jaccard's coefficient. Traditional graph partitioning algorithms, on the other hand, use the structure of a graph to find highly connected components (subgraphs) (e.g. [1]). Given a graph $G = (V, E)$ the algorithms assign the vertices V to a set of k partitions (clusters) in such a way that prescribed properties such as minimum cutsize or maximum connectivity are optimized. Graph partitioning techniques were developed for use on graphs where most of the information is contained in the structure—edge and node weights are the only attribute information that is considered.

This paper will examine adaptations to an existing spectral graph partitioning technique to consider both link structure and attribute information. Graph partitioning techniques are often designed to operate on $V \times V$ matrix of edge weights. The general goal is to partition the graph such that weights within clusters are maximized and weights between clusters are minimized. These techniques can be used for data clustering problems providing the data are represented as an $N \times N$ matrix of similarity scores (entry n_{ij} is the similarity of instances i and j). In this situation, the data forms a complete graph—every pair of instances has a weighted edge between them. We will follow this approach, and consider a number similarity metrics that incorporate both link structure and attribute information into the weight matrix.

3 Spectral Clustering

Spectral clustering originated in the 70s with graph partitioning techniques that exploited the connection between eigenvectors and algebraic properties of a graph [9, 11]. Although, finding an optimal partition is in general NP complete, the eigenvector corresponding to the second smallest eigenvalue of the *Laplacian* matrix of the graph provides some information that can be used to guide an approximate solution. Experimental evidence has shown this heuristic approach to often work well in practice. Recently Shi and Malik presented a new clustering algorithm that uses spectral partitioning to optimize a *normalized cut* objective function (see equation 1) [21]. This technique has been successfully applied in a number of domains, including image segmentation [21] and document clustering [8], and has prompted further investigation into the properties of spectral clustering techniques. We investigate the application of this algorithm to relational domains. Specifically, we will investigate the design of similarity metrics that can incorporate the two sources of information and explore the characteristics that underlie successful metrics.

3.1 Algorithm

Spectral partitioning algorithms cluster weighted graphs through eigenvalue decomposition of a weighted adjacency matrix. We base our approach on the *normalized cut* spectral partitioning algorithm introduced by Shi and Malik for image segmentation [21]. The algorithm is a divisive, hierarchical clustering algorithm, which takes a graph $G = (V, E)$, a set of k attributes $\mathbf{A} = \{A^1, \cdots, A^k\}$, where $A^k = \{a_i^k : v_i \in V\}$, and a similarity function S, where $S(i, j)$ defines the similarity between $v_i, v_j \in V$, and recursively partitions the

graph, minimizing the normalized cut objective function. We outline the algorithm below and describe the similarity metrics in section 3.4:

- *Input*: G, $\mathbf{A}$, S, m, c
- *Algorithm*:
  1. Let $\mathbf{W}$ be an $N \times N$ matrix with $\mathbf{W}_{ij} = S(i,j)$.
  2. Let $\mathbf{D}$ be an $N \times N$ diagonal matrix with $d_i = \sum_{j \in V} S(i,j)$.
  3. Solve the eigensystem $(\mathbf{D} - \mathbf{W})\mathbf{x} = \lambda \mathbf{D}\mathbf{x}$ for the eigenvector $\mathbf{x}_1$ associated with the second smallest eigenvalue λ_1.
  4. Sort $\mathbf{x}_1$.
  5. Consider m evenly spaced entries in $\mathbf{x}_1$. For each value x_{1m}:
     (a) Bipartition the nodes into (A, B) such that $A \cap B = \emptyset$ and $A \cup B = V$ and $x_{1a} < x_{1m}\ \forall v_a \in A$.
     (b) Calculate the normalized cut objective function $J(A,B)$:

$$J(A,B) = \frac{cut(A,B)}{\sum_{i \in A} d_i} + \frac{cut(A,B)}{\sum_{j \in B} d_j}$$
$$\text{where } \ cut(A,B) = \sum_{i \in A, j \in B} S(i,j) \tag{1}$$

  6. Partition the graph into the (A, B) that minimizes J.
  7. Calculate the *stability*[1] of the current cut, if $stability > c$ stop recursing.
  8. Recursively repartition A and B if necessary.

In general, it takes $O(n^3)$ operations to solve for all eigenvalues of an arbitrary eigensystem. However, if the weight matrix is sparse, the Lanczos algorithm can be used to compute the solution in $O(n^{1.4})$ operations [20, 7], and approximate algorithms can compute the solution in $O(|E|)$ operations [14]. Similarity metrics that produce sparse matrices are preferable for this reason.

3.2 Algorithm Correctness

The algorithm outlined above is an approximate solution that minimizes the normalized cut criterion; finding an optimal solution is, in general, an NP-hard problem [21]. Shi and Malik [21] have shown that when there is a partition (A, B) of V such that:

$$\mathbf{x}_{1i} = \begin{cases} \alpha, & i \in A \\ \beta, & i \in B \end{cases} \tag{2}$$

then (A, B) is the optimal partition—it minimizes the normalized cut criterion. Furthermore, the value of the cut itself is equal to λ_1.

This result indicates that when the eigenvector $\mathbf{x}_1$ is *piecewise linear* with respect to a partition (A, B), the algorithm will correctly identify the partition. Recent analysis has focused on achieving a more thorough understanding of the algorithm's performance. For example, under what conditions will $\mathbf{x}_1$ be piecewise linear?, and what is the impact on algorithm performance as piecewise linearity degrades? We will examine these questions

[1] We use the stability calculation and threshold proposed in [21]. The eigenvector values are binned into m evenly spaced bins; the stability value is the ratio of the size of the bin with the minimum number of values, to the size of the bin with the maximum number of values. This measure is intended to stop partitioning when the distribution of eigenvector values is too uniform—indicating that the eigenvector is far from being piecewise linear. All the experiments in this paper used the settings: $m = \lceil log_2(N) + 1 \rceil$, and $c = 0.06$.

as we investigate a number of similarity metrics, exploring the characteristics which lead to superior performance.

Meila and Shi [17] connect spectral clustering to Markov random walks and provide a characterization of the cases in which the normalized cut method will be exact. We outline the relevant portions of their work in propositions 1-3 below (see [17] for proofs). Next, we examine the asymptotic correctness of the algorithm, analyzing the similarity metric as a random variable, and show that the eigenvectors will become piecewise linear as cluster size grows. Analysis of higher-level partitioning is complicated by the recursive nature of the algorithm so we restrict our consideration to the (simpler) case of a single bipartitioning of the graph.

Proposition 1*: If* λ, $\mathbf{x}$ *are solutions of* $(\mathbf{D}-\mathbf{W})\mathbf{x} = \lambda\mathbf{D}\mathbf{x}$, *and* $\mathbf{P} = \mathbf{D}^{-1}\mathbf{W}$, *then* $(1-\lambda)$, $\mathbf{x}$ *are solutions of* $\mathbf{P}\mathbf{x} = \lambda\mathbf{x}$.

Here $\mathbf{P}$ is the normalization of the weight matrix to a stochastic matrix. This shows the equivalence of the spectral problem formulated for normalized cut and the eigenvectors/values of the stochastic matrix $\mathbf{P}$. We will analyze the correctness of the spectral algorithm using $\mathbf{P}$.

Proposition 2*:* $\mathbf{P}$ *has an eigenvector that is piecewise constant w.r.t. a partition* $\Delta = (A_1, A_2)$ *of* V, *with a non-zero eigenvalue, if and only if the sums* $\mathbf{P}_{is} = \sum_{j \in A_s} \mathbf{P}_{ij}$ *are constant for all* $i \in A_{s'}$ *and all* $A_s, A_{s'} \in \Delta$, *and the matrix* $\mathbf{R} = [\mathbf{P}_{ss'}]_{s,s'=1,2}$ *is non-singular, where* $\mathbf{P}_{ss'} = \sum_{j \in A_{s'}} \mathbf{P}_{ij}$, *for any* $i \in A_s$.

This shows that a stochastic matrix has piecewise constant eigenvectors if the underlying Markov random walk can be viewed as a Markov chain with state space $\Delta = (A_1, A_2)$ and transition probability matrix $\mathbf{R}$. We refer to this property as *block-stochastic*. This shows how spectral clustering groups nodes based on the similarity of their transition probabilities to subsets of the graph.

Proposition 3*: If* $\mathbf{P}$ *is block-stochastic, and the eigenvalues of* $\mathbf{R}$ *are larger than the spurious eigenvalues of* $\mathbf{P}$, *then the bipartition of* V *is exact.*

This shows that the spectral algorithm requires more that just piecewise constant eigenvectors to produce an exact bipartition. The largest eigenvalue of $\mathbf{R}$ is equivalent to the largest eigenvalue of $\mathbf{P}$—the associated eigenvector corresponds to an (uninteresting) partitioning which groups the entire graph in a single component. It is easy to see in this case that the transition probabilities are constant (at 1.0) across all the nodes in the cluster. The 2^{nd} largest eigenvalue of $\mathbf{R}$ identifies the eigenvector that is used in the spectral algorithm. If this eigenvalue is larger than the other $n-2$ eigenvalues of $\mathbf{P}$, then by the algorithm will recover the partition Δ exactly.

3.3 Asymptotic Analysis

Propositions $1-3$ show a set of conditions under which the spectral algorithm will return an exact partitioning. This analysis does not show, however, algorithm behavior when the cluster transition probabilities are no longer constant. Empirical evidence indicates that the algorithm will find good partitions even when the transition probabilities are far from constant. Ideally, we would like to characterize the conditions necessary for optimal performance and bound algorithm performance otherwise. As a first step, we investigate the effect of intra- and inter-cluster transition probabilities on algorithm performance, analyzing the asymptotic performance as the distributions converge to constants. Section 4 reports empirical experiments that explore finite sample behavior.

In particular, we consider the impact of the similarity metric on piecewise linearity of the eigenvectors. Using the law of large numbers, we show that in the limit, as $|A_1|, |A_2| \rightarrow$

∞, a similarity metric with distinguishable intra- and inter-cluster means, will produce a nearly piecewise linear eigenvector. If the associated eigenvalue is larger than the spurious eigenvalues of $\mathbf{P}$, the algorithm will identify the exact partition.

***Proposition 4**: Let $\Delta = (A_1, A_2)$ be a partition of V. Let the function $S(i,j)$ define the similarity measure between $v_i, v_j \in V$. If, $\forall i,j,k$, $S(i,j)$ is conditionally independent of $S(i,k)$ given node i, and $E[\mathbf{R}_{11}]E[\mathbf{R}_{22}] \neq E[\mathbf{R}_{12}]E[\mathbf{R}_{21}]$ then, $\mathbf{P}$ has an eigenvector that will converge to piecewise constant w.r.t. Δ as $|A_1|, |A_2| \to \infty$.*

Proof. In order to simplify the calculations below, we assume that the two clusters share the same distribution of intra- and inter- cluster similarity values. Let μ_{in} be the mean intra-cluster similarity for nodes $i,j \in A_1$ or $i,j \in A_2$. Similarly, let μ_{out} be the mean inter-cluster similarity for nodes $i \in A_1$ and $j \in A_2$.

We can represent each entry in $\mathbf{W}$ as a random variable. Consider the entries of row i. The entries $\mathbf{W}_{ij}, \mathbf{W}_{ik}$ are not independent because the similarity values are both based on node i. However, conditioned on the state of i (e.g. attribute values of i), the entries can be viewed as independent random variables if the state of j is independent of the state of k. This assumption corresponds to a generative model in which the objects and links in the graph are conditionally independent given the object cluster memberships.

We will calculate the expected intra- and inter-cluster transition probabilities in $\mathbf{P}$ as a ratio of sums of random variables. Let T^i_{in} be the total intra-cluster transition probability for node i, where $i \in A_{k,k\in 1,2}$, and let $|A_k| = n_k$. Similarly, let T^i_{out} be the total inter-cluster transition probability, and T^i_{all} be the total transition probability. Then $\mathbf{P}^i_{in}$ is the ratio of T^i_{in} and T^i_{all}, and $\mathbf{P}^i_{out}$ is the ratio of T^i_{out} and T^i_{all}.

Analytical derivations of the mean and variance of $\mathbf{P}^i_{in}$ and $\mathbf{P}^i_{out}$ are included in the appendix, we report only the relevant details here. When $S(i,j)$ is conditionally independent of $S(i,k)$ given the state of node i, the cluster transition probabilities are simply sums of independent random variables:

$$\begin{aligned} E[T^i_{in}] &= n_k \cdot \mu_{in} \\ E[T^i_{out}] &= n_{k'} \cdot \mu_{out} \\ E[T^i_{all}] &= (n_k \cdot \mu_{in}) + (n_{k'} \cdot \mu_{out}) \end{aligned} \tag{3}$$

The normalized transition probabilities in $\mathbf{P}$ then correspond to the ratio of two random variables (e.g., T^i_{in}/T^i_{all}), which can be approximated using a truncated Taylor series expansion. The expectation and variance for intra- and inter-cluster normalized transition probabilities are as follows:

$$\begin{aligned} E[\mathbf{P}^i_{in}] &= E[T^i_{in}/T^i_{all}] &&\approx \frac{\mu_{T_{in}}}{\mu_{T_{all}}} \cdot [1 + [\frac{\sigma_{T_{all}}}{\mu_{T_{all}}}]^2 - \frac{\sigma_{T_{in}T_{all}}}{\mu_{T_{in}}\mu_{T_{all}}}] \\ E[\mathbf{P}^i_{out}] &= E[T^i_{out}/T^i_{all}] &&\approx \frac{\mu_{T_{out}}}{\mu_{T_{all}}} \cdot [1 + [\frac{\sigma_{T_{all}}}{\mu_{T_{all}}}]^2 - \frac{\sigma_{T_{out}T_{all}}}{\mu_{T_{out}}\mu_{T_{all}}}] \end{aligned} \tag{4}$$

where σ_{XY} is the covariance of X, Y.

As $n_1, n_2 \to \infty$, it follows directly from the *Law of Large Numbers* [3] that the value of $T^i_{in}/T^j_{in} \to 1$ for $i,j \in A_k$, since T_{in} is a sum of independent random variables with finite mean and variance. A similar argument holds for T_{out} and T_{all}. Now consider the normalized transition probabilities for $\mathbf{P}$. If, in the limit, the sums T^i_{in} (and T^i_{out}, T^i_{all}) converge to the same value for all $i \in A_k$, then the normalized sums $\mathbf{P}^i_{in}$ will converge to the same value $\mathbf{P}_{in}$ for all $i \in A_k$. A similar argument holds for $\mathbf{P}^i_{out}$.

As $n_1, n_2 \to \infty$, we can decompose the matrix $\mathbf{P}$ into $\mathbf{P} = \mathbf{P}' + \epsilon\mathbf{E}$, where $\mathbf{P}'$ is a matrix with constant transition probabilities $\mathbf{P}_{in}$ and $\mathbf{P}_{out}$, and $\mathbf{E}$ is a perturbation matrix with $||\mathbf{E}||_2 = 1$. Then by standard matrix perturbation theory [12]:

$$\begin{aligned} &(\mathbf{P}' + \epsilon\mathbf{E})\mathbf{x}_i(\epsilon) = \lambda_i(\epsilon)\mathbf{x}_i(\epsilon) \\ &where \;\; \mathbf{x}_i(\epsilon) = \mathbf{x}_i + \epsilon \textstyle\sum_{j=1, j\neq i}^{n} \{ \frac{\mathbf{y}_j{}^T \mathbf{E}\mathbf{x}_i}{(\lambda_i - \lambda_j)\mathbf{y}_j{}^T\mathbf{x}_i} \} + O(\epsilon^2) \;, \\ &and \quad \lambda_i(\epsilon) = \lambda_i \pm \tfrac{\epsilon}{|\mathbf{y}_i{}^T\mathbf{x}_i|} \end{aligned} \tag{5}$$

Here $\mathbf{x}_i$, $\mathbf{y}_i$, and λ_i, are the right and left eigenvectors, and the eigenvalues of $\mathbf{P}'$. As $n_1, n_2 \to \infty$, $\epsilon \to 0$ and the eigenvectors of $\mathbf{P}$ will converge to the eigenvectors of $\mathbf{P}'$. Therefore the graph will converge to a Markov chain with state space $\Delta = (A_1, A_2)$, and constant transition probabilities $\mathbf{R}_{11} = \mathbf{R}_{22} = E[\mathbf{P}^i_{in}]$, and $\mathbf{R}_{12} = \mathbf{R}_{21} = E[\mathbf{P}^i_{out}]$. If $\mathbf{R}_{11} \neq \mathbf{R}_{12}$, then $\mathbf{R}$ will be non-singular, and by proposition 2, $\mathbf{P}$ will have a piecewise linear eigenvector w.r.t Δ. □

This analysis shows that any similarity metric will produce piecewise linear eigenvectors in the limit, provided the intra- and inter-cluster means not equal, and $S(i, j)$ is conditionally independent of $S(i, k)$ given i, for all i, j, k. Furthermore, if the eigenvalue of interest is greater than the spurious eigenvalues, then the algorithm will find the exact partition of V, in the limit. This analysis indicates that all metrics will perform equally as sample sample goes to infinity. We expect however, that finite sample performance will vary based on the characteristics of the metrics. In particular, we expect that performance will be influenced by the mean and variance of the intra- and inter cluster transition probabilities. We present a number of metrics below and investigate their finite sample performance in section 4 to identify the situation in which a metric can be expected to perform well.

3.4 Similarity Metrics

In order to adapt conventional spectral partitioning algorithms to relational data, we need to define a similarity metric that incorporates both link structure and attribute information. The previous section shows that all similarity metrics will perform equivalently in the limit, but this does not guarantee comparable finite sample performance. We define five similarity functions below and evaluate their performance on a variety of synthetic and real datasets in the following sections.

3.4.1 Attribute Information Only

We refer to this metric as *AttrOnly*. It calculates the similarity between objects i and j by examining each of k attributes on the two objects and counting the number of attribute values they have in common.

$$\begin{aligned} &S(i, j) = \tfrac{1}{k} \textstyle\sum_k s_k(i, j) \\ &\text{where} \;\; s_k(i, j) = \begin{cases} 1 & \text{if } k_i = k_j \\ 0 & \text{otherwise} \end{cases} \end{aligned} \tag{6}$$

This metric is generally known as the matching coefficient. We scale the value by the total number of attributes so the maximum similarity between two objects is 1. The similarity measure is used to weight all pairs of nodes in V. This metric ignores the link information in the graph. It considers the objects as if they formed a complete graph and consequently, efficient eigensolver techniques are not applicable. This metric is included as a baseline

conventional clustering technique. We expect this approach to work well when the attribute values are highly correlated with the cluster membership.

3.4.2 Link Information Only

We refer to this metric *LinkOnly*. It calculates the similarity between objects i and j by examining the edges of G. Objects that are directly related by an edge in the graph have a similarity of 1 and objects that are not directly related have a similarity of 0.

$$S(i,j) = \begin{cases} 1 & \text{if } e_{ij} \in E \text{ or } e_{ij} \in E \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

This metric uses the link structure alone, attribute values are not considered. This metric is included as a second baseline, this time for a conventional graph partitioning technique that considers an unweighted graph in isolation. We expect this approach to work well when the graph edges are highly correlated with cluster membership. Since relational data graph are sparse in general, efficient eigensolver techniques can be used with this metric.

3.4.3 Link Information Included as Additional Attribute

This is the first *hybrid* metric that incorporates both attribute and link information. We refer to this metric as *LinkAsAttr*. It calculates the similarity between objects i and j in the same manner as *AttrOnly*—by examining each of k attributes on the two objects and counting the number of attribute values they have in common. The difference is that the links of the graph are incorporated into the metric as the $(k+1)^{th}$ attribute.

$$S(i,j) = \begin{cases} \frac{1}{k+1}\left(\sum_k s_k(i,j)\right) + 1\right) & \text{if } e_{ij} \in E \text{ or } e_{ij} \in E \\ \frac{1}{k+1}\sum_k s_k(i,j) & \text{otherwise} \end{cases}$$
$$\text{where } \; s_k(i,j) = \begin{cases} 1 & \text{if } k_i = k_j \\ 0 & \text{otherwise} \end{cases} \tag{8}$$

This approach is perhaps the most obvious way to include link information in a metric that matches attribute values in some manner. With no prior knowledge of the domain, we have no reason to expect that the link structure contains more information than the attribute value. However, the link structure is often central in relational domains—for example, in a graph of hyperlinked web documents, we expect a link to confer more information about topic clustering than a match on a single word for two pages. The next metric is designed to capture this intuition.

3.4.4 Weighted Combination of Link and Attribute Information

We refer to the second hybrid metric as *WtLinkAttr*. It calculates the similarity between objects i and j by a weighted combination of the *LinkOnly* metric and the *AttrOnly* metric.

$$S(i,j) = \begin{cases} \frac{1}{2}\left(\frac{1}{k} \cdot \sum_k s_k(i,j) + c\right) & \text{if } e_{ij} \in E \text{ or } e_{ij} \in E \\ \frac{1}{2}\left(\frac{1}{k} \cdot \sum_k s_k(i,j) + 0\right) & \text{otherwise} \end{cases}$$
$$\text{where } \; s_k(i,j) = \begin{cases} 1 & \text{if } k_i = k_j \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

This metric considers the link structure and attributes as equal sources of information. A measure close to one indicates that the nodes share an edge, or have a number of attributes

in common. This metric should capture more of the information in a sparse relational graph—two nodes in the same cluster should have similar attribute values, but they may not have a direct edge between them (even though they should be indirectly linked through common neighbors).

When $c = 1$, we refer to this metric as *WtLinkAttr1*. This metric combines the link and attribute information uniformly. The sparsity of relational graphs, however, will cause the expected intra-cluster link similarity to be less than one, even if the links are perfectly correlated with cluster membership. In this case, if the link and attribute information are combined uniformly, the attribute noise may drown out a strong link signal. An approach that gives the link information proportionally more weight (e.g., $c > 1$) may achieve better performance.

In practice we will not know how to scale the link information to contribute an amount equal to the attribute information, but for the synthetic experiments discussed in the next section we know the maximum edge probability is 0.2, so we can scale the link weights by setting $c = 5$, which makes the maximum expected value 1. We refer to this metric as *WtLinkAttr2*. Although we will not know the scaling factor in practice, we include this metric to help illuminate the differences between *WtLinkAttr1* and the metric below. Specifically, we include this metric to test the conjecture that the poor performance of *WtLinkAttr2* is due to the relatively weak link signal being combined uniformly with the attribute signal.

3.4.5 Filter Attribute Information Through Links

We refer to the last hybrid metric as *LinkAsFilter*. It calculates the similarity between objects i and j by looking at the edges of G, then by examining each of k attributes on the two objects, and counting the number of attribute values they have in common. Objects that are not directly related by an edge in the graph have a similarity of 0 regardless of their attribute values.

$$S(i,j) = \begin{cases} \frac{1}{k}\sum_k s_k(i,j) & \text{if } e_{ij} \in E \text{ or } e_{ij} \in E \\ 0 & \text{otherwise} \end{cases}$$
$$\text{where } \; s_k(i,j) = \begin{cases} 1 & \text{if } k_i = k_j \\ 0 & \text{otherwise} \end{cases} \tag{10}$$

This metric uses the *AttrOnly* similarity measure to weight the existing edges of graph G. This approach maintains the sparsity of the relational data graph so the algorithm can use efficient eigensolver techniques. (The previous hybrid techniques do not have this property.) At the same time, it incorporates both sources of information into the metric. A measure close to one indicates that the nodes share an edge as well as a number of attributes in common.

4 Synthetic Data Experiments

In order to identify the situations where we can expect each of the similarity metrics to perform well, we evaluate algorithm performance on synthetic data sets for which the correct clustering is known. This facilitates analysis over a wide range of conditions.

The goal of this work is to use attribute and link information to improve clustering results. The implicit assumption is that an approach using both sources of information will do better than an approach using either source in isolation. To evaluate this claim, we record performance of the *AttrOnly* metric (equation 6), which uses attribute information in iso-

lation, and the *LinkOnly* metric (equation 7), which uses link information in isolation. We compare these results to four hybrid metrics that combine link and attribute information: *LinkAsFilter* (equation 10), *LinkAsAttr* (equation 8), *WtLinkAttr1* and *WtLinkAttr2* (equation 9).

From this study we show that the *LinkAsFilter* and *WtLinkAttr2* metrics achieve high accuracy for a broad spectrum of datasets with varying link and attribute characteristics. Furthermore, we show the mechanism by which these metrics combine the link and attribute information to achieve superior performance.

4.1 Synthetic Data

Our synthetic data sets are comprised of undirected, unipartite, connected graphs ($G = (V, E)$) where nodes corresponds to objects and edges correspond to relations among the objects. Each graph contains 200 nodes unless otherwise indicated. A binary attribute, $C = \{+, -\}$ is used to represent cluster membership. Cluster labels are assigned randomly to each object with $P(+) = 0.5$. Each object has five binary attributes, where the attribute values are assigned randomly given the objects cluster label (e.g., $P(A_5 = 1|C = +) = 0.9$). Cluster labels determine which edges (links) are added to the graph as well. Each pair of objects in V are considered independently and an edge is added randomly given the cluster labels of the two objects (e.g., $P(e_{ij} \in E|C_i = C_j) = 0.18$).

The experiments record algorithm performance while varying both attribute and link association. To measure the effect of attribute information on performance, the experiments varied the strength of the relationship between the attribute values and the cluster label. Within each level of correlation, all five attributes were generated with the same probability:

$$\begin{aligned} P(A = 1|C = +) &= \{0.50, 0.55, \ldots, 0.95, 1.0\} \\ P(A = 1|C = -) &= 1.0 - P(A = 1|C = +) \end{aligned} \tag{11}$$

The symmetry in cluster attribute parameters simplifies the analytical analysis but it is not necessary for algorithm correctness. To measure the effect of link information on performance, the experiments also varied the strength of the relationship between the link structure and the cluster label. Intra-cluster and inter-cluster links were generated with the following range of probabilities:

$$\begin{aligned} P(e_{ij}|C_i = C_j) &= \{0.10, 0.12, \ldots, 0.18, 0.20\} \\ P(e_{ij}|C_i \neq C_j) &= 0.2 - P(e_{ij}|C_i = C_j) \end{aligned} \tag{12}$$

Here the symmetry, and range of probabilities, was chosen to produce a graph with approximately 10% of the $n(n-1)/2$ possible edges. This level of linkage is comparable to the levels of sparsity we have observed in real-world relational data sets.

4.2 Metric Performance

The first experiment measures the accuracy of the six metrics across the range of attribute and link probabilities described above. We report the accuracy of the clusterings returned by each similarity metric, averaged over 100 trials at the same setting. Note that the bottom, foremost corner represents completely random link and attribute information so no metric should do better than 0.5 at that point.

Figure 1 shows the results of this experiment where attribute and link correlations are varied simultaneously. These graphs show expected results for the *LinkOnly* and *AttrOnly* metrics. When the attribute signal is moderate to high the *AttrOnly* metric performs well, but poorly otherwise, and the *LinkOnly* metric performs similarly with respect to the link

signal. The *LinkAsAttr* and *WtLinkAttr1* metrics achieve performance comparable to the *AttrOnly* metric. This is due to the method of incorporating link and attribute information into the metric. Because the link signal is relatively weak ($P(e_{ij}|C_i = C_j) \leq 0.2$), random attribute information drowns out the link information if the two sources of information are combined uniformly (e.g., *WtLinkAttr1*) or if the attribute information is given proportionally more weight (e.g., *LinkAsAttr*).

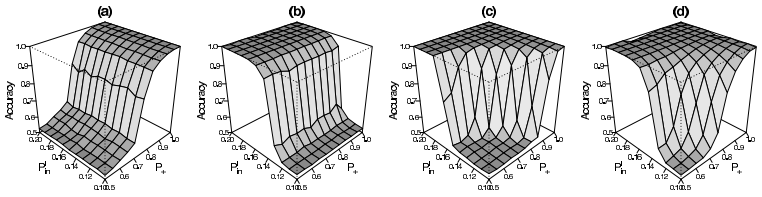


Figure 1: Cluster accuracy on synthetic data: (a) *AttrOnly* metric, (b) *LinkOnly* metric, (c) *LinkAsAttr*, (d) *WtLinkAttr1*, and (e) *WtLinkAttr2*, and (f) *LinkAsFilter*.

The results show that the *LinkAsFilter* and *WtLinkAttr2* metrics achieve near-perfect accuracy over the widest range of conditions, with *LinkAsFilter* covering more of the space than *WtLinkAttr2*. These two metrics are able to combine the link and attribute information successfully and should yield good results in datasets where either the link or the attributes are moderately correlated with the clusters (as well as cases where both are correlated). This indicates that the link signal needs to be weighted appropriately and incorporated in equal parts to the attribute signal, in order to improve the performance of the algorithm with the *WtLinkAttr1* metric.

The *LinkAsFilter* and *WtLinkAttr2* metrics do not always perform as well as the *LinkOnly* and *AttrOnly* metrics. This illustrates the tradeoff for utilizing both sources of information—the additional information increases variance and will result in decreased performance for some situations, in exchange for better coverage of the space of possible dataset characteristics. In particular, consider the *LinkOnly* results where the association with the cluster is moderate. When the link correlation is moderate and the attribute correlation is low, both hybrid metrics achieve significantly lower accuracy than would be achieved considering only links in isolation. Similar behavior is apparent for the *AttrOnly* metric, when the attribute correlation is moderate and the link correlation is low. However, notice that the effect is more pronounced in this situation. This indicates that the two metrics rely more heavily on link information than attribute information.

4.3 Algorithm Analysis

It is clear that the *LinkAsFilter* and *WtLinkAttr2* metrics achieve superior performance over a wider range of data characteristics, but we would like to understand the mechanism through which the metrics affect algorithm performance. For example, why does *LinkAsFilter* differ from *WtLinkAttr2*? Will *LinkAsFilter* always be preferable to *WtLinkAttr2*? How can we extend this work to combine a third source of information (e.g., temporal extent)?

Following our analysis in section 3.2, we hypothesize that the metrics affect algorithm performance through their distributions of intra- and inter- cluster similarity transition probabilities (e.g., equations 5). As the total intra- and inter- cluster transition probabilities converge to constants in the limit, we know that the existence of a piecewise linear eigenvector is guaranteed and all metrics should perform equivalently regardless of the data characteristics. However, we observe a wide range of performance at relatively small samples, both across data characteristics and across metrics. This indicates that the performance

difference may be due to a difference in the statistical power of the various metrics. In particular, asymptotic analysis shows that the algorithm can distinguish among clusters with arbitrarily small differences in mean transition probability as long as variance goes to zero (in the limit). In finite samples, the clusters will be distinguishable if the distributions of their intra-/inter-cluster transition probabilities are *separable*—where separation depends on the mean and standard deviation of the intra-/inter-cluster similarity measures.

Given our data generation parameters, we can calculate intra- and inter-cluster mean similarity scores analytically. As an example, we include the mean derivations for the *LinkAsAttr* metric. Let $p_{A_i} = P(A = 1|C = C_i)$, $p_{\neg A_i} = P(A = 0|C = C_i)$, $p_{l_{ij}} = P(e_{ij}|C_i = C_j)$, and $p_{\neg l_{ij}} = P(e_{ij}|C_i \neq C_j)$. Then the expected similarity value for *LinkAsAttr* is as follows:

$$\begin{aligned} E[S(i,j)] = \quad & \sum_{a=0}^{k} \sum_{b=0}^{1} \binom{k}{a} p_{m_{ij}}^{a} p_{\neg m_{ij}}^{k-a} p_{l_{ij}}^{b} p_{\neg l_{ij}}^{1-b} \cdot \frac{a+b}{k+1} \\ where \quad & p_{m_{ij}} = (p_{A_i} p_{A_j} + p_{\neg A_i} p_{\neg A_j}) \\ & p_{\neg m_{ij}} = (p_{A_i} p_{\neg A_j} + p_{\neg A_i} p_{A_j}) \end{aligned} \tag{13}$$

Means for the other similarity metrics, and variances, are calculated in the same manner. With these parameters, we can compute means and variances for cluster transition probabilities (e.g., equations 5). Recall that our data generation process produces the same distribution for each cluster—each cluster has the same intra- and inter-cluster similarities—so we can examine a single set of distributions, $\mu_{P_{in}} = E[\mathbf{P}_{in}]$ and $\mu_{P_{out}} = E[\mathbf{P}_{out}]$. Furthermore, we know that the transition probabilities in $\mathbf{P}$ are normalized to sum to one. This means we can simply examine $\mu_{P_{in}}$, instead of examining the separation between $\mu_{P_{in}}$ and $\mu_{P_{out}}$. A value of $\mu_{P_{in}} = 1.0$ corresponds to maximum separation between the two clusters, and a value of $\mu_{P_{in}} = 0.5$ corresponds to no separation.

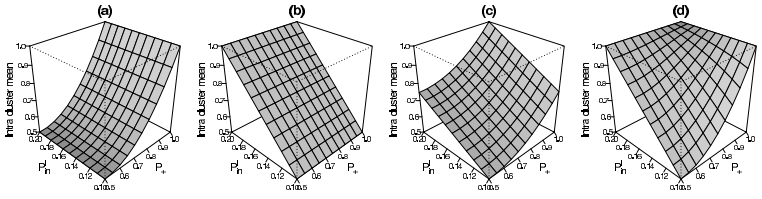


Figure 2: Intra-cluster means on synthetic data: (a) *AttrOnly* metric, (b) *LinkOnly* metric, (c) *LinkAsAttr*, (d) *WtLinkAttr1*, (e) *WtLinkAttr2*, and (f) *LinkAsFilter*.

Figure 2 graphs the results of the first experiment again, but this time plots $\mu_{P_{in}}$ vs. attribute and link correlations. There are a number of important observations to draw from these results. First, the shapes of the graphs are very similar to the accuracy graphs in figure 1. This indicates that there is a strong relationship between mean separation and algorithm performance. Second, the areas where we observe perfect performance (e.g., accuracy $= 1.0$) do not necessarily correspond to maximum mean separation (e.g., $\mu_{P_{in}} \leq 1.0$). This illustrates the difference between the *LinkAsFilter* and *WtLinkAttr2* metrics. We will discuss this in more detail below. Figure 3d graphs a box plot of $\mu_{P_{in}}$ for each metric individually—the edges of the boxes represent the 25^{th} and 75^{th} percentiles, the middle line of the box corresponds to the median, and the extreme points correspond to minimum and maximum values. This is a one-dimensional summary of the data in figure 2, which again illustrates that the $\mu_{P_{in}}$ is significantly higher for the *LinkAsFilter* metric on average.

To examine the effect of $\mu_{P_{in}}$ on algorithm performance, we analyze the data from all metrics concurrently. Figure 3a graphs $\mu_{P_{in}}$ vs. accuracy for the experiments reported

above, including all the metrics in the same graph[2]. There is a clear relationship between $\mu_{P_{in}}$ and algorithm performance, with a correlation of 0.849 ($p < 2e-16$)—accuracy is consistenly high for $\mu_{P_{in}} > 0.7$ and consistently low for $\mu_{P_{in}} < 0.6$. Figure 3b graphs $\mu_{P_{in}}$ vs. accuracy as well, but for these experiments we increased the number of objects in the graph to 500. This illustrates the effect of decreasing variance—in this case the threshold of $\mu_{P_{in}} = 0.65$ (the vertical line in the figure) is a good predictor of algorithm performance. Figure 3c graphs $\mu_{P_{in}}$ vs. accuracy as well, but for these experiments we decreased the number of objects in the graph to 50. This illustrates the effect of increasing variance—although the correlation is still high (0.797), it reduces the clear behavior we see in the the previous two figures. These plots show that algorithm performance is affected by both the mean and variance of the transition probabilities.

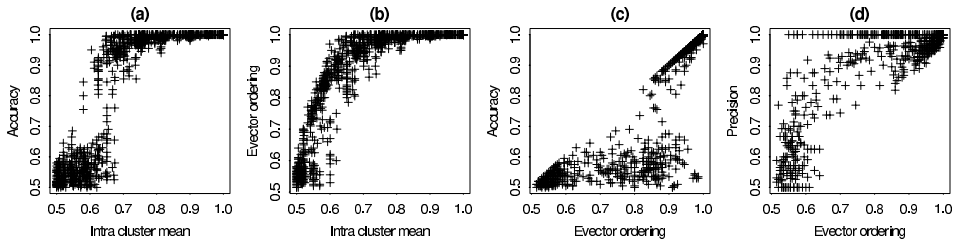


Figure 3: Analysis of intra-cluster mean on algorithm performance: (a) 200 objects, (b) 500 objects, (c) 50 objects, and (d) distribution of mean per metric for 3a.

To understand the mechanism by which $\mu_{P_{in}}$ affects algorithm performance, we looked at the relationship between $\mu_{P_{in}}$ and the eigenvector values in $\mathbf{x}_1$ using three different measures. The first measure is the stability metric described in section 3.1, which measures the piecewise linearity of the eigenvector values. This is graphed in figure 4a—the stability values exhibit increased variance as mean separation decreases, but there is no clear difference between $\mu_{P_{in}} < 0.7$ and $\mu_{P_{in}} > 0.7$. The second measure records the maximum gap between any two consecutive eigenvector values (in the sorted eigenvector). This is another measure of the piecewise linearity of the eigenvector—if the objects are ordered correctly, the gap should correspond to the separation of the two clusters in the eigenvector. This measure is graphed in figure 4b—there appears to be a non-linear relationship between the gap value and $\mu_{P_{in}}$, but it clearly distinguishes only very high values of $\mu_{P_{in}}$. The last measure is intended to measure the quality of the ordering in the (sorted) eigenvector. The linear search for an optimal partition should not be adversely affected by degradation of piecewise linearity unless the degradation also affects the ordering of objects' eigenvector values. To measure this, we looked at the sorted eigenvector and the set of m possible partition values considered by the algorithm. We recorded the maximum accuracy achieved by any of those partitions. If the maximum accuracy is low, this indicates disorder in the eigenvector. This measure is graphed in figure 4c—it shows that decreasing $\mu_{P_{in}}$ results in a disordering of the eigenvector values. The few outliers at $\mu_{P_{in}} = 1.0$ correspond to a small number of *LinkAsFilter* trials, where the metric disconnects the graph. This will be discussed more below.

The relationship between $\mu_{P_{in}}$ and eigenvector ordering exhibits the same behavior as the relationship between $\mu_{P_{in}}$ and accuracy. For $\mu_{P_{in}} > 0.7$, there is little disorder in the eigenvector values. Figure 5a shows the relationship between eigenvector ordering and algorithm performance. There are two effects in this graph—there is a clear relationship between eigenvector ordering and accuracy, however, there are a significant number of tri-

[2]For graphing purposes, we include only a random sample of 5 observations for each metric, at each level of attribute and link correlation.

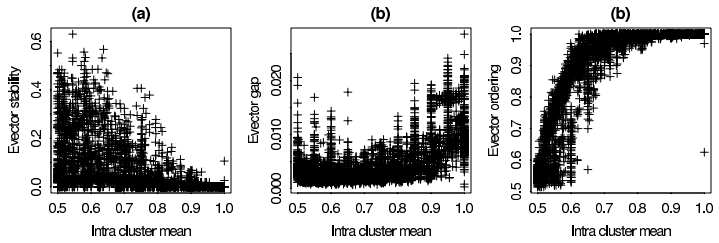


Figure 4: Analysis of intra-cluster mean on eigenvector values: (a) piecewise linearity of the values, (b) maximum gap between consecutive values, and (c) proportion of objects correctly ordered in the eigenvector.

als with very little disorder but still low accuracy. This effect is explained by figures 5b-c, where we graph the precision of the smallest cluster chosen by the algorithm. This shows that when the algorithm achieves low accuracy, it is often because a small, but pure, cluster is broken off from the rest of the graph. The density plot shows that the algorithm generally finds a small cluster of high precision, regardless of metric or data condition. Furthermore, the precision doesn't degrade until there is a high level of disorder in the eigenvector[3]. It is unclear why the algorithm breaks off small, high-precision clusters even when the eigenvector ordering is correct. This is not a spurious effect due to the algorithm only considering a small number of thresholds (e.g., m evenly-spaced points). It remains consistent even when we set $m = N$. However, it only appears in three of the hybrid metrics: *LinkAsAttr*, *WtLinkAttr1*, and *WtLinkAttr2*. Further investigation is needed to determine the interaction between the three metrics and the normalized cut objective function in these cases.

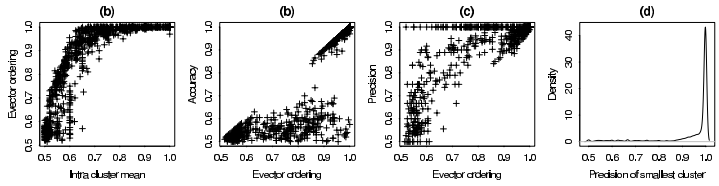


Figure 5: Analysis of eigenvector ordering on algorithm performance: (a) accuracy overall, (b) precision of smallest cluster, and (c) distribution of precision for all trials.

This analysis shows that mean separation affects algorithm performance through the ordering of the objects' eigenvector values, but how does variance interact with mean separation to degrade performance? Figure 6 graphs the performance of the six metrics for samples of size 50. Compare this to figure 1 to see that performance degradation is not uniform across metrics. The *LinkOnly* and *LinkAsFilter* metrics are adversly affected over a wider range of data conditions. For example, notice that *WtLinkAttr2* is now superior, or at least comparable, to *LinkAsFilter*. This illustrates the primary distinction between *LinkAsFilter* and *WtLinkAttr2*. The *LinkAsFilter* metric reduces the amount of information it uses in order to increase the mean separation between the clusters. Because it is filtering the attribute information through the existing edges of the graph, it throws away both useful and noisy data and increases the variance of the transition probabilities. If the sample size is large enough to withstand this increase in variance, then the metric will produce superior clusterings.

[3]Recall that these graphs include trials where the link and attribute structure are both uncorrelated with cluster membership, so there are a number of points where we can't expect the algorithm to perform better than random.

However, when the sample size is low the filter can do more harm than good. For example, filtering through the existing edges may disconnect a previously connected cluster. In these situations, it may be best to use the *WtLinkAttr2* metric, which suffers less from increased variance, but performs well over a small range of data characteristics. This indicates that expected sample size may influence our choice of metric. However, since we do not know how to set c *WtLinkAttr2* in practice, and because it produces a sparse similarity matrix, we focus on the *LinkAsFilter* metric in our empirical data experiments, reported in the next section.

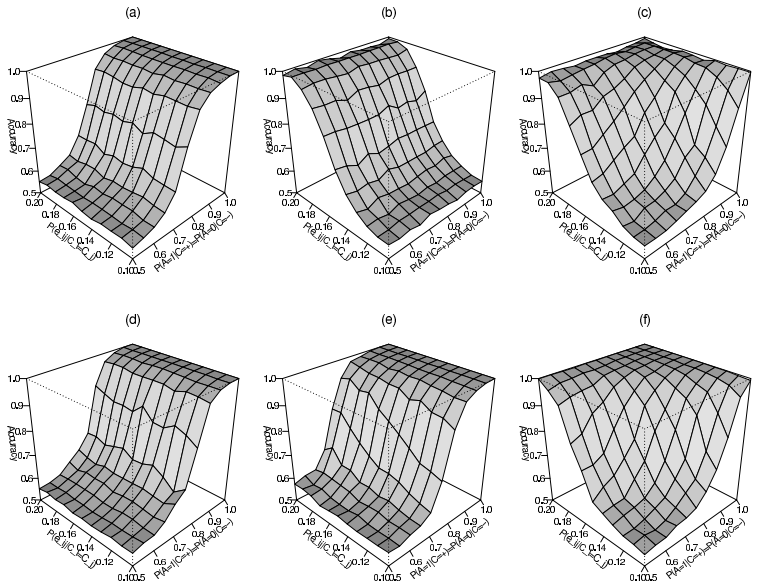


Figure 6: Accuracy on smaller datasets (size 50): (a) *AttrOnly* metric, (b) *LinkOnly* metric, (c) *LinkAsFilter*, (d) *LinkAsAttr*, (e) *WtLinkAttr1*, and (f) *WtLinkAttr2*.

5 Empirical Experiments

The experiments reported below are intended to evaluate two assertions. The first claim is that the *LinkAsFilter* clustering approach can be used to find groups of items with similar attribute values and high inter-connectedness. We evaluate this claim by comparing the clusters produced by the *LinkAsFilter* metric to randomly generated clusters of the same size, evaluating intra-cluster attribute similarity and intra-cluster linkage.

The second claim is that the *LinkAsFilter* clustering approach finds meaningful clusters. It is a difficult task to evaluate clusterings of datasets for which there is no right answer [2]. One approach is to present the resulting clusters for user examination. For this type of subjective evaluation, we include example cluster members from two real-world datasets. Another, more objective, approach is to examine cluster utility by evaluating the cluster labels ability to improve a related classification task. We evaluate three approaches (*LinkOnly*, *AttrOnly*, and *LinkAsFilter*) on a third real-world dataset in this manner, and show the *LinkAsFilter* clusters achieve a significant improvement in classification accuracy.

5.1 Datasets

We applied our clustering techniques to three real-world datasets where attributes exhibit correlation among linked objects, and the link structure exhibits clustering. These are the characteristics we expect to find in datasets that contain communities, and it is in these situations that we expect our clustering algorithms will perform well.

The first data set is drawn from Cora, a database of computer science research papers extracted automatically from the web using machine learning techniques [16]. We selected the largest connected component from the set of machine-learning papers published after 1993. The resulting graph contains 1,042 papers and 2546 citation links. The similarity metric considered two topic attributes at different levels of granularity (e.g. Machine Learning/Planning/etc., and Neural Networks/Rule Learning/etc.).

The second data set consists of a set of web pages from four computer science departments, collected by the WebKB Project [6]. The web pages have been manually classified into the categories: course, faculty, staff, student, research project, or other. The category "other" denotes a page that is not a home page (e.g. a curriculum vitae linked from a faculty page or homework description linked from a course page). The collection contains approximately 4,000 web pages and 8,000 hyperlinks among those pages. We clustered the largest connected component in these data—a graph of 1236 pages and 3673 hyperlinks. The similarity metric considered two attributes: page category and department.

The third data set is a relational data set containing information about the yeast genome at the gene and the protein level[4]. The data set contains information about 1,243 genes and 1,734 interactions among their associated proteins. We clustered the largest connected component, which consisted of 814 genes and 1475 interactions. The similarity metric considered 13 boolean function attributes (each gene may have multiple functions). We evaluated the resulting cluster labels ability to predict gene localization. We applied a relational Bayesian classifier [19] to the entire dataset, using the cluster labels as an additional attribute, and measured change in accuracy.

5.2 Results

For the WebKB, Cora, and Gene datasets we report results using the *LinkAsFilter* similarity metric, described in equation 10, in the recursive clustering algorithm described in section 3.1. All empirical experiments evaluated $\lceil m = logN + 1 \rceil$ possible partitions, and used a stopping (stability) threshold of 0.06 empirically determined by Shi and Malik [21].

Clustering the sample of Cora papers produced 71 clusters varying in size from 1-202 papers. We report statistics for the 28 clusters with more than six papers. The number of papers in each cluster is shown in figure 7(c). Table 1 includes randomly selected titles from four clusters for subjective evaluation. Although we did not use title words in the similarity metrics, the clusters show a surprising uniformity among the titles. This indicates that research papers can be clustered into meaningful groups using the citation structure and topic attributes alone.

Figure 7(a) shows the actual and expected proportion of intra-cluster citations. The expected proportions were calculated as the average proportions over ten random clusterings. For all but the largest cluster, the proportion of intra-cluster citations is significantly higher than the expected values. This indicates that the clustering technique is finding groups of highly inter-connected research papers.

To evaluate intra-attribute similarity we averaged the attribute similarity across all pairs of pages within each cluster. Again as a baseline measure we calculated the average attribute

[4] www.cs.wisc.edu/ dpage/kddcup2001/

Table 1: Cora cluster examples

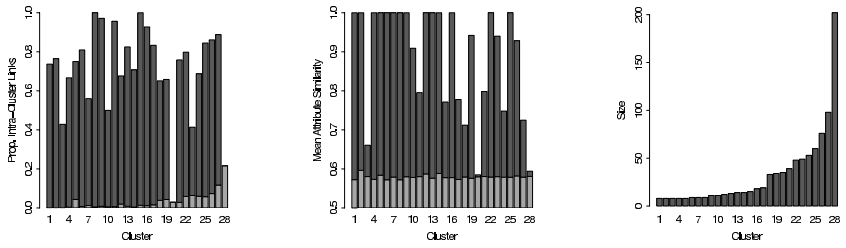
Cluster 9: Belief revision: A critique; Plausibility measures and default reasoning; Modeling belief in dynamic systems. Part I: foundations; Knowledge-Based Framework for Belief Change, Part II: Revision and Update; Iterated revision and minimal revision of conditional beliefs; An event-based abductive model of update; On the logic of iterated belief revision; A unified model of qualitative belief change: A dynamical systems perspective; Generalized update: Belief change in dynamic settings
Cluster 14: In defense of C4.5: Notes on learning one-level decision trees; Exploring the decision forest: An empirical investigation of Occams razor in decision tree induction; Algorithmic stability and sanity-check bounds for leave-one-out cross-validation; Bias and the quantification of stability; Characterizing the generalization performance of model selection strategies; A new metric-based approach to model selection; Preventing overfitting of Cross-Validation data; Further experimental evidence against the utility of occams razor
Cluster 19: An empirical evaluation of bagging and boosting; On-line portfolio selection using multiplicative updates; Heterogeneous uncertainty sampling for supervised learning; Improved boosting algorithms using confidence-rated predictions; On-line algorithms in machine learning; Training algorithms for hidden Markov models using entropy based distance functions; A system for multiclass multi-label text categorization; Coevolutionary Search Among Adversaries
Cluster 24: Refinement of Bayesian networks by combining connectionist and symbolic techniques; DistAl: An inter-pattern distance-based constructive learning algorithm; An Anytime Approach to Connectionist Theory Refinement: Refining the Topologies of Knowledge-Based Neural Networks; Creating advice-taking reinforcement learners; Learning controllers for industrial robots; Generating accurate and diverse members of a neural-network ensemble; A Neural Architecture for a High-Speed Database Query System; Comparing methods for refining certainty-factor rule-bases;

similarity between pages in ten random clusterings. Figure 7(b) plots the intra-cluster attribute similarity compared to the expected averages given random clusterings. Most clusters show a higher than expected attribute similarity. The largest cluster, however, does not exhibit significantly high linkage or attribute similarity—this set of papers may contain the set of papers that could not be partitioned into smaller clusters (i.e., the papers with no coherent community structure).

Clustering the sample of WebKB pages produced 55 clusters varying in size from 1-649 pages. We report statistics for the 15 clusters with more than six pages. The number of pages in each cluster is shown in figure 8(c). Table 2 includes randomly selected URLs from four clusters for subjective evaluation. The selected clusters are primarily populated by pages from the University of Wisconsin. Furthermore the clusters appear to group by function—for example, tech reports, course pages, or research group pages.

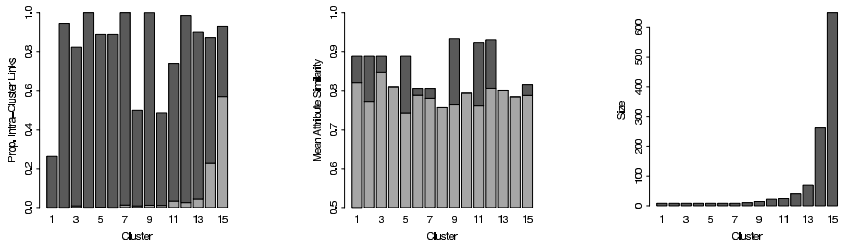
We analyzed the link structure and attribute similarity of each cluster to evaluate the results. Figure 8(a) shows the actual and expected proportion of intra-cluster hyperlinks. For all clusters, the proportion of intra-cluster citations is significantly higher than the expected values. The exception is the largest cluster, which only has slightly higher than expected linkage. Again, this may indicate that the largest cluster contains the set of pages that do not contain coherent communities. Figure 8(b) plots the intra-cluster averages compared to the expected averages given random clusterings. This dataset does exhibit significantly higher than expected attribute similarity. However, the algorithm is still able to cluster pages into groups that are highly inter-connected. This indicates that the *LinkAsFilter* metric may be robust to irrelevant attribute values.

Clustering the sample of genes produced 88 clusters varying in size from 1-140 genes. We report statistics for the 14 clusters with more than six genes. The number of genes in each cluster is shown in figure 9(c). Figure 9(a) shows the actual and expected propor-



(a) (b) (c)

Figure 7: Evaluation of hybrid clusters for Cora dataset.



(a) (b) (c)

Figure 8: Evaluation of hybrid clusters for WebKB dataset.

tion of intra-cluster citations. For all clusters, the proportion of intra-cluster citations is significantly higher than the expected values. Figure 9(b) plots the intra-cluster attribute similarity compared to the expected averages given random clusterings. These results show a moderate amount of attribute similarity within clusters. Again, the largest cluster appears to contain the set of genes that could not be grouped into smaller clusters effectively.

Genes do not have interpretable identifiers, such as URLs or titles, to present for subjective evaluation. However, the structure of genomic data offers an opportunity for a more objective evaluation of the clustering results. Clusters of inter-connected genes with similar associated functions, may indicate a group of genes that are interacting to perform a particular function in the cell. If this is the case, the cluster labels should be helpful to predict gene localization in the cell. To test this hypothesis, we used the cluster labels in a relational classification task. The learning task was to predict a genes localization in the cell. There are 15 values for localization, including nucleus and cell wall.

The first set of experiments, reported in figure 10, compare the performance of the *LinkOnly*, *AttrOnly*, *LinkAsFilter* metrics. We report average 10-fold cross-validation accuracies for RBC models learned using the cluster labels from each each metric. This shows that the cluster labels alone are not very good predictors of gene localization, although *AttrOnly* and *LinkAsFilter* perform slightly better than *LinkOnly*.

The second set of experiments compare the three metrics when the other attributes in the data are incorporated into the models. The baseline RBC model used twelve attributes for prediction, including gene phenotype, motif, and interaction type, and achieved an average

Table 2: WebKB cluster examples

Cluster 5: http://www.cs.wisc.edu/Dienst/UI/2.0/Describe/ncstrl.uwmadison/CS-TR-89-890; http://www.cs.wisc.edu/Dienst/UI/2.0/Describe/ncstrl.uwmadison/CS-TR-90-947; http://www.cs.wisc.edu/Dienst/UI/2.0/Describe/ncstrl.uwmadison/CS-TR-95-1283; http://www.cs.wisc.edu/Dienst/UI/2.0/Describe/ncstrl.uwmadison/CS-TR-91-1037; http://www.cs.wisc.edu/Dienst/UI/2.0/Describe/ncstrl.uwmadison/CS-TR-90-962; http://www.cs.wisc.edu/Dienst/UI/2.0/Describe/ncstrl.uwmadison/CS-TR-89-900; http://www.cs.wisc.edu/~reps/reps.html; http://www.cs.wisc.edu/Dienst/UI/2.0/Describe/ncstrl.uwmadison/CS-TR-91-1038
Cluster 9: http://www.cs.wisc.edu/~bart/537/quizzes/quiz6.html; http://www.cs.wisc.edu/~bart/cs537.html; http://www.cs.wisc.edu/~bart/537/quizzes/quiz3.html; http://www.cs.wisc.edu/~bart/537/quizzes/quiz10.html; http://www.cs.wisc.edu/~bart/537/quizzes/quiz2.html; http://www.cs.wisc.edu/~bart/537/programs/program2.html; http://www.cs.wisc.edu/~bart/537/lecturenotes/titlepage.html; http://www.cs.wisc.edu/~bart/537/quizzes/quiz9.html;
Cluster 11: http://www.cs.wisc.edu/~cs354-2/cs354/lec.notes/numbers.html; http://www.cs.wisc.edu/~cs354-2/cs354/lec.notes/data.structures.html; http://www.cs.wisc.edu/~cs354-2/cs354/solutions/Q2.j.html; http://www.cs.wisc.edu/~cs354-2/cs354/lec.notes/arch.features.html; http://www.cs.wisc.edu/~cs354-2/cs354/lec.notes/interrupts.html; http://www.cs.wisc.edu/~cs354-2/cs354/lec.notes/case.studies.html; http://www.cs.wisc.edu/~cs354-2/cs354/lec.notes/arith.int.html; http://www.cs.wisc.edu/~cs354-2/cs354/lec.notes/MAL.html;
Cluster 14: http://www.cs.wisc.edu/ condor/ research.html; http://www.cs.wisc.edu/ ~bart/ cs638.html; http://www.cs.wisc.edu/ coral/ coral.people.html; http://www.cs.wisc.edu/ ~brad/ brad.html; http://www.cs.wisc.edu/ ~sastry/ spring96.html; http://www.cs.wisc.edu/ ~ashraf/ ashraf.html; http://maf.wisc.edu/ distributed/ condor/ index.html; http://www.cs.wisc.edu/ ~ssl/ resume.html;

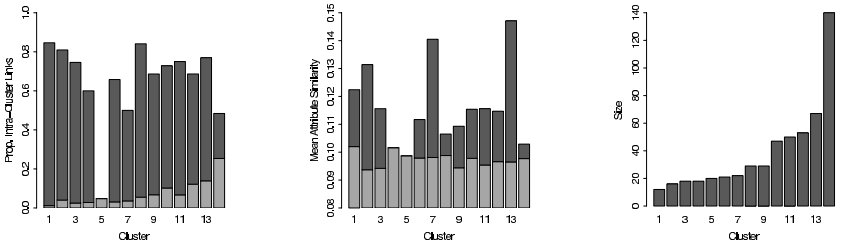
accuracy of 66.3%. The RBC model that included cluster labels from the *AttrOnly* technique did not significantly[5] improve accuracy. However, the model that included cluster labels from the *LinkOnly* technique achieved an average accuracy of 68.4%, a significant improvement. This indicates that gene interactions alone are helpful for predicting location. The model that included cluster labels from the *LinkAsFilter* technique achieved an average accuracy of 70.2%. This is a significant improvement over both the *LinkOnly* model and the baseline RBC model without cluster labels. This shows that gene *communities*, identified by the *LinkAsFilter* technique, can improve classification models of gene localization, over and above clustering based on attributes or links in isolation.

6 Related Work

There has been relatively little work investigating clustering techniques for relational domains. The work in this area has focused on either complex generative models with latent variables (e.g. PRMs), or augmented clustering techniques that use ad-hoc similarity metrics to incorporate both link and attribute information.

Probabilistic models have been developed to model cluster membership using both attribute information and link structure. Cohn and Hoffman [5] outline a generative model where a documents topic determines both its content and its citations. The model without link structure (content only) is known as probabilistic latent semantic indexing (pLSI). To our

[5]We assessed the significance of results using two-tailed, paired t-tests over the ten-fold cross-validation trials, with $\alpha = 0.05$. The null hypothesis is that there is no difference in the accuracy between two approaches; the alternative is that there is a significant difference in performance.



(a) (b) (c)

Figure 9: Evaluation of hybrid clusters for Gene dataset.

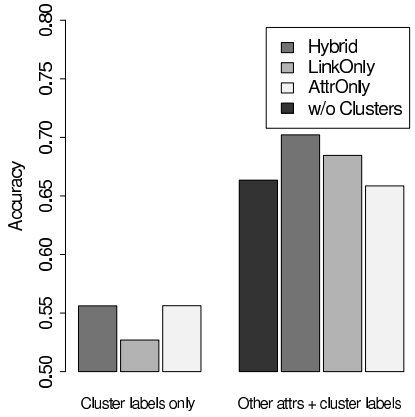


Figure 10: Accuracy predicting gene location with cluster labels.

knowledge, the Cohn and Hoffman model has not been evaluated in a clustering context. Kubica, Moore, Schneider, and Yang [15] propose a probabilistic model of link structure based on cluster membership. The model considers both attribute information and link structure but combines them in an alternative manner. In the generative model, attributes determine group membership and group membership determines the link structure. Taskar, Segal and Koller [23] use probabilistic relational models (PRMs) to cluster relational data with attribute and links. PRMs are directed graphical models, which can be used to cluster for hidden group variables. However, the models acyclicity constraint makes it difficult to apply to network data with complex dependencies.

HyPursuit [24] was the first information retrieval system to cluster documents using semantic information in both document contents and hyperlink structure. The system defined a complex similarity metric to capture both content and link structure correspondence between pages. The hybrid similarity metric can be used with any conventional clustering algorithm because it is defined over all pairs of pages. It is difficult to assess the utility of the metric however, because evaluation consists of a subjective assessment on a single clustering task.

Modha and Spangler [18] also propose an algorithm for clustering hypertext documents using document contents and hyperlink structure. The authors capture three features of

documents in their new similarity measure: (1) word similarity, (2) out-link similarity, and (3) in-link similarity. A geometric hypertext-clustering algorithm is used, which extends the classical Euclidean k-means algorithm [10]. Modha and Spangler include parameters to control the influence of the three features. They include a search for the optimal parameter setting in their algorithm but do not evaluate the impact of different settings. They do note that several settings were chosen across clustering experiments. This indicates that different web graphs may contain varying levels of textual and link information.

He, Ding, Zha, and Simon [13] use a spectral graph-partitioning algorithm to automatically identify topics in sets of retrieved web pages. This approach is quite similar to our spectral approach, however He et al. use a different similarity measure designed for high-dimensional text domains. In addition, they augment the hyperlink graph with weighted co-citation links. The algorithm automatically clusters query result sets for topics and presents the user with the most authoritative pages from each topic.

7 Discussion and Conclusions

This paper presents a spectral clustering algorithm that exploits both attribute information and link structure to improve clustering of relational data. It is intuitively plausible that link structure can be combined with attribute information to effectively group relational data. However, there has been relatively little work investigating clustering techniques for relational domains. Due to the efficiency of probabilistic relational models with latent variables, we chose to explore extensions to recent spectral clustering techniques for relational data.

To encourage a deeper understanding of the design of similarity metrics, which incorporate multiple sources of information, we explore the characteristics that underlie successful similarity metrics. This is where we differ from previous work in hybrid clustering algorithms. We have set up a framework to evaluate different similarity metrics quantitatively over a wide range of relational data sets, and in the context of a spectral clustering algorithm. Our experiments show that increasing the separation between total intra-cluster and inter-cluster transition probabilities results in superior performance over a wide range of data characteristics. Metrics that drop potentially noisy information from consideration (e.g. *LinkAsFilter*) increase this separation, but there must be enough data to withstand the associated increase in variance. An additional advantage of metrics that dismiss noisy information is algorithm efficiency—there are $O(E)$ approximate eigensolver algorithms and $O(n^{1.4})$ exact eigensolver algorithms for sparse matrices.

We have analyzed the spectral decomposition algorithm from a statistical perspective and show that the successful hybrid metrics use the link and attribute information to increase the separation between noisy clusters. We have shown an empirical connection between the distribution of transition probabilities and algorithm performance, connecting both mean and variance to cluster accuracy. Future work will attempt to derive theoretical bounds on finite-sample performance, and explore the interaction between the normalized cut metric and accuracy at low mean separations.

Acknowledgments

This research is supported under a AT&T Graduate Research Fellowship and by DARPA and AFRL, AFMC, USAF under contract numbers F30602-00-2-0597 and F30602-01-2-0566. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright notation hereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements either expressed or implied, of the DARPA, the AFRL or the U.S. Government.

References

[1] C. Alpert and A. Kahng. Recent directions in netlist partitioning: A survey. *VLSI Journal*, 1995.

[2] P. Arabie, L. Hubert, and G. DeSoete. *Clustering and Classification*. World Scientific, 1996.

[3] L. Bain and M. Engelhardt. *Introduction to Probability and Mathematical Statistics*. Druxbury, 1992.

[4] F. Chung. *Spectral Graph Theory*. The American Mathematical Society, 1997.

[5] D. Cohn and T. Hofmann. The missing link - a probabilistic model of document content and hypertext connectivity. *Advances in Neural Information Processing Systems*, 10, 2001.

[6] M. Craven, D. DiPasquo, D. Freitag, A. McCallum, T. Mitchell, K. Nigam, and S. Slattery. Learning to extract symbolic knowledge from the world wide web. In *Proceedings of the 15th National Conference on Artificial Intelligence*, 1998.

[7] J. Cullum and R. Willoughby. *Lanczos Algorithms for Large Symmetric Eigenvalue Computations, Vol. 1*. Birkhäuser, 1985.

[8] I. Dhillon. Co-clustering documents and words using bipartite spectral graph partitioning. In *Proc. of the 7th ACM International Conf. on Knowledge Discovery and Data Mining*, 2001.

[9] W. Donath and A. Hoffman. Lower bounds for the partitioning of graphs. *IBM Journal of Research and Development*, 17):420–425, 1973.

[10] everitt. *Cluster Analysis*. John Wiley and Sons, Inc, 1993.

[11] M. Fiedler. Algebraic connectivity of graphs. *Czecheslovak Math. Jour.*, 23(98):298–305, 1973.

[12] G. Golub and C. Van Loan. *Matrix Computations*. Johns Hopkins University Press, 1983.

[13] X. He, C. Ding, H. Zha, and H. Simon. Automatic topic identification using webpages clustering. In *Proceedings of the 1st IEEE International Conference on Data Mining*, 2001.

[14] R. Kannan, S. Vempala, and A. Vetta. On clusterings: Good, bad and spectral. In *Proceedings of the 41st Symposium on the Foundations of Computer Science*, 2000.

[15] J. Kubica, A. Moore, J. Schneider, and Y. Yang. Stochastic link and group detection. In *Proceedings of the 18th National Conference on Artificial Intelligence*, 2002.

[16] A. McCallum, K. Nigam, J. Rennie, and K. Seymore. A machine learning approach to building domain-specific search engines. In *Proceedings of the 16th International Joint Conference on Artificial Intelligence*, 1999.

[17] M. Meila and J. Shi. A random walks view of spectral segmentation. In *Proceedings of the 8th International Workshop on Artificial Intelligence and Statistics*, 2001.

[18] D. Modha and W. Spangler. Clustering hypertext with applications to web searching. In *Proceedings of the 11th ACM Conference on Hypertext and Hypermedia*, 2000.

[19] J. Neville, D. Jensen, and B. Gallagher. Simple estimators for relational bayesian classifiers. In *Proceedings of the 3rd IEEE International Conference on Data Mining*, 2003.

[20] B. Parlett. *The Symmetric Eigenvalue Problem*. Prentice-Hall, Inc., 1980.

[21] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

[22] H. Small and B. Griffith. The structure of scientific literatures i: Identifying and graphing specialties. *Science Studies*, 4:17–40, 1974.

[23] B. Taskar, E. Segal, and D. Koller. Probabilistic clustering in relational data. In *Proceedings of the 17th International Joint Conference on Artificial Intelligence*, 2001.

[24] R. Weiss, B. Velez, M. Sheldon, C. Namprempre, P. Szilagyi, A. Duda, and D. Gifford. Hypursuit: A hierarchical network search engine that exploits content-link hypertext clustering. In *Proceedings of the ACM Conference on Hypertext*, 1996.

Appendix

The expectation for T^i_{in}, is calculated using conditional expectation[6] on the state of i, which we refer to as i_S:

$$\begin{aligned} E[T^i_{in}] &= E[\textstyle\sum_{j\in A_k} S(i,j)] \\ &= \textstyle\sum_{i_S} p(i_S) \cdot E[\sum_{j\in A_k} S(i_S,j)] \\ &= \textstyle\sum_{i_S} p(i_S) \cdot n_k \cdot E[S(i_S,j)|j \in A_k] \\ &= n_k \cdot \textstyle\sum_{i_S} p(i_S) \cdot \sum_{j_S} p(j_S) \cdot S(i_S,j_S) \\ &= n_k \cdot \textstyle\sum_{i_S} \sum_{j_S} p(i_S) \cdot p(j_S) \cdot S(i_S,j_S) \\ &= n_k \cdot E[S_{in}] \\ &= n_k \cdot \mu_{in} \end{aligned} \tag{14}$$

Total inter-cluster and overall means are calculated in a similar fashion:

$$\begin{aligned} E[T^i_{out}] &= n_{k'} \cdot \mu_{out} \\ E[T^i_{all}] &= (n_k \cdot \mu_{in}) + (n_{k'} \cdot \mu_{out}) \end{aligned} \tag{15}$$

The variance of the total intra-cluster similarity is calculated as follows[7]:

$$\begin{aligned} Var[T^i_{in}] &= Var[\textstyle\sum_{j\in A_k} S(i,j)] \\ &= E_{i_S}\{Var[\textstyle\sum_{j\in A_k} S(i_S,j)]\} \\ &= \textstyle\sum_{i_S} p(i_S) \cdot Var[\sum_{j\in A_k} S(i_S,j)] \\ &= \textstyle\sum_{i_S} p(i_S) \cdot n_k \cdot Var[S(i_S,j)|j \in A_k] \\ &= n_k \cdot \textstyle\sum_{i_S} \sum_{j_S} p(i_S) \cdot p(j_S) \cdot \{S(i_S,j_S) - E_{i_S}[S(i_S,j_S)]\}^2 \end{aligned} \tag{16}$$

Total inter-cluster and overall variance are calculated in a similar fashion:

$$\begin{aligned} Var[T^i_{out}] &= n_{k'} \cdot \textstyle\sum_{i_S} p(i_S) \cdot Var[S(i_S,j)|j \in A_{k'}] \\ Var[T^i_{all}] &= \textstyle\sum_{i_S} p(i_S) \left\{n_{k'} Var[S(i_S,j)|j \in A_{k'}] + n_k Var[S(i_S,j)|j \in A_k]\right\} \end{aligned} \tag{17}$$

From these we can calculate the expected transition probabilities of $\mathbf{P}$ using the ratio of two random variables (e.g., T_{in}/T_{all})[8]. The expectation and variance for intra- and inter-cluster normalized transition probabilities are as follows:

---

[6] The derivation of the mean uses the theorem: $E[h(X,Y)] = E_X\{E[h(X,Y)|X]\}$ [3]

[7] The derivation of the variance uses the following equivalence:

$$\begin{aligned} Var(h(X,Y)) &= E[h(X,Y)^2] - E[h(X,Y)]^2 \\ &= E_X\{E[h(X,Y)^2|X]\} - E_X\{E[h(X,Y)|X]^2\} \\ &= E_X\{Var(h(X,Y)|X)\} \end{aligned}$$

[8] These calculations use an approximation of the ratio of two random variables, based on a truncated Taylor series expansion (cite?):

$$\begin{aligned} E[X/Y] &\approx \tfrac{\mu_X}{\mu_Y} \cdot [1 + [\tfrac{\sigma_Y}{\mu_Y}]^2 - \tfrac{\sigma XY}{\mu_X \mu_Y}] \\ Var(X/Y) &\approx [\tfrac{\mu_X}{\mu_Y}]^2 \cdot [[\tfrac{\sigma_X}{\mu_X}]^2 + [\tfrac{\sigma_Y}{\mu_Y}]^2 - 2\tfrac{\sigma XY}{\mu_X \mu_Y}] \end{aligned}$$

$$
\begin{aligned}
E[\mathbf{P}^i_{in}] &= E[T^i_{in}/T^i_{all}] &&\approx \tfrac{\mu_{T_{in}}}{\mu_{T_{all}}} \cdot [1 + [\tfrac{\sigma_{T_{all}}}{\mu_{T_{all}}}]^2 - \tfrac{\sigma_{T_{in}T_{all}}}{\mu_{T_{in}}\mu_{T_{all}}}] \\
Var[\mathbf{P}^i_{in}] &= Var[T^i_{in}/T^i_{all}] &&\approx [\tfrac{\mu_{T_{in}}}{\mu_{T_{all}}}]^2 \cdot [[\tfrac{\sigma_{T_{in}}}{\mu_{T_{in}}}]^2 + [\tfrac{\sigma_{T_{all}}}{\mu_{T_{all}}}]^2 - 2\tfrac{\sigma_{T_{in}T_{all}}}{\mu_{T_{in}}\mu_{T_{all}}}] \\
E[\mathbf{P}^i_{out}] &= E[T^i_{out}/T^i_{all}] &&\approx \tfrac{\mu_{T_{out}}}{\mu_{T_{all}}} \cdot [1 + [\tfrac{\sigma_{T_{all}}}{\mu_{T_{all}}}]^2 - \tfrac{\sigma_{T_{out}T_{all}}}{\mu_{T_{out}}\mu_{T_{all}}}] \\
Var[\mathbf{P}^i_{out}] &= Var[T^i_{out}/T^i_{all}] &&\approx [\tfrac{\mu_{T_{out}}}{\mu_{T_{all}}}]^2 \cdot [[\tfrac{\sigma_{T_{out}}}{\mu_{T_{out}}}]^2 + [\tfrac{\sigma_{T_{all}}}{\mu_{T_{all}}}]^2 - 2\tfrac{\sigma_{T_{out}T_{all}}}{\mu_{T_{out}}\mu_{T_{all}}}]
\end{aligned}
\tag{18}
$$

where σ_{XY} is the covariance of X, Y. For the equations above, the covariance of T_{in} and T_{all} reduces to the variance of T_{in}, using conditional expectation to eliminate the covariance. A similar derivation applies to the covariance of T_{out} and T_{all}.

$$
\begin{aligned}
\sigma_{T_{in}T_{all}} &= E[T_{in}T_{all}] - E[T_{in}] \cdot E[T_{all}] \\
&= E[T_{in}(T_{in} + T_{out})] - E[T_{in}] \cdot E[(T_{in} + T_{out})] \\
&= E[T^2_{in} + T_{in} \cdot T_{out}] - E[T_{in}]^2 - E[T_{in}] \cdot E[T_{out}] \\
&= E[T^2_{in}] + E[T_{in} \cdot T_{out}] - E[T_{in}]^2 - E[T_{in}] \cdot E[T_{out}] \\
&= E[T^2_{in}] - E[T_{in}]^2 + E[T_{in} \cdot T_{out}] - E[T_{in}] \cdot E[T_{out}] \\
&= Var(T_{in}) - \textstyle\sum_{i_S} p(i_S)\{E[T_{in} \cdot T_{out}|i] - E[T_{in}|i] \cdot E[T_{out}|i]\} \\
&= Var(T_{in}) - \textstyle\sum_{i_S} p(i_S) \cdot 0 \\
&= Var(T_{in})
\end{aligned}
$$

□