

Correlation and Sampling in Relational Data Mining

David Jensen and Jennifer Neville
{jensen|jneville}@cs.umass.edu
Knowledge Discovery Laboratory
Computer Science Department
Univ. of Massachusetts
Amherst, MA 01003 USA

Abstract

Data mining in relational data poses unique opportunities and challenges. In particular, *relational autocorrelation* provides an opportunity to increase the predictive power of statistical models, but it can also mislead investigators using traditional sampling approaches to evaluate data mining algorithms. We investigate the problem and provide new sampling approaches that correct the bias associated with traditional sampling.

1. Introduction

We are studying how to learn predictive models from relational data with PROXIMITY, a knowledge discovery system for analyzing very large relational data structures. In this paper, we report on issues of correlation and sampling that arise from the special features of relational data. Specifically, we identify a common type of probabilistic dependence in relational data that we call *relational autocorrelation*. We show how this type of autocorrelation can bias evaluations of data mining algorithms that use traditional approaches to sample relational data. We describe and evaluate new sampling procedures that can prevent this bias.

We discuss these findings in greater detail below. First, we review current uses of relational data and describe the object-and-relational-neighborhood (ORN) approach to relational learning. Next, we demonstrate and describe relational autocorrelation. We then describe a family of sampling procedures that can be used to evaluate the effects of relational autocorrelation and give the results of applying those procedures in a complex relational data set. Finally, we briefly describe planned future work.

2. Objects and Relational Neighborhoods

In this paper, we examine how to construct and evaluate classification models in *relational* data. Relational data consist of objects (representing people, places, and things) connected by links (representing persistent relationships among objects). Often, the objects and links are heterogeneous, representing many different types of real-world entities and relationships.

For example, relational data could be used to represent information about the motion picture industry, where objects represent studios, movies, awards, and persons (e.g., actors, directors, and producers) and links represent relationships (e.g., actor-in and remake-of). A schema for one such movie database from the UCI repository is shown in Figure 1.

This type of relational data can be stored in conventional relational databases, or in less structured formats such as graph databases, web page repositories, or semi-structured databases. In the case of PROXIMITY, we have implemented a specialized graph database on top of a conventional relational database engine.

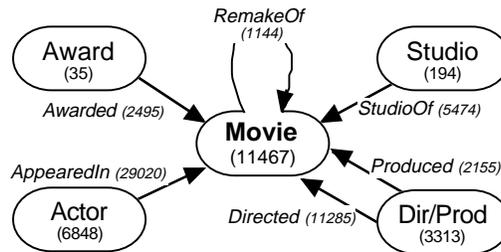


Figure 1: *The movie database schema. Attributes on objects and links are not shown. Numbers indicate the frequency of each element in the UCI movie data.*

Our approach to data mining in relational data focuses on objects and their relational neighborhoods. Our techniques build models that predict the value of an attribute of an object based on *intrinsic attributes* of that object and *relational attributes* that capture characteristics of the “relational neighborhood” of the object. For example, we have constructed statistical models that predict the genre of a movie (e.g., drama, comedy, suspense) based on its intrinsic attributes (e.g., year of production) and its relational attributes (e.g., total number actors). A relational attribute can capture either the characteristics of a single related object (e.g., years of experience of a movie’s director) or the aggregate characteristics of multiple objects in the relational neighborhood (e.g., average years of experience of all a movie’s actors). We call this approach to relational learning the *object and relational neighborhood* (ORN) approach.

The general idea of ORN is a common feature underlying many systems for relational learning. For example, probabilistic relational models (Friedman et al. 1999) use field values from individual database records and aggregations of values from related records to construct Bayesian networks. WEBKB (Craven et al. 1998) learns naïve Bayes classifiers and rules in first-order logic that employ intrinsic attributes of webpages and their links to other pages to classify the pages into categories (e.g., student homepage, departmental homepage, and course homepage). Several methods from inductive logic programming (Muggleton 1992) use intrinsic and relational clauses to make predictions about individual objects. Finally, our own prior work on PROXIMITY uses an ORN approach to classify objects based on intrinsic and relational attributes (Neville and Jensen 2000). These approaches differ widely in the expressiveness of their model representation and the methods used to construct models. Still, all construct models that predict the attributes of an object based on the attributes of that object and those of closely related objects.

3. An Example: Classifying Movies

We have successfully learned classification models using this basic framework of objects and their local neighborhoods, but relational data can pose unique opportunities and challenges that are not typically encountered in traditional data mining problems. As an example, consider the problem of constructing a classification model to predict the *genre* of a movie (e.g., *drama*, *comedy*, *suspense*) based on a movie’s actors, director, producer, studio, and awards. Specifically, we calculated nine relational attributes of movies (e.g., $AwardCount(x)$, $HasRemake(x)$, $ActorCount(x)$, and $PercentWomenActors(x)$), and used them to construct naïve Bayes classifiers to predict *genre*. We randomly generated five different pairs of training and test sets, where each pair contains disjoint sets of movies.

For each pair, we constructed classifiers on the training set, estimated their accuracy on the test set, and then averaged the five trials.¹

The classifier is able to achieve an average test-set accuracy of 40%. Accuracy for the best default classifier (*genre = drama*) is 24%, and the standard deviation of the estimate is less than 0.007. While the classifier is far from perfect, it appears to represent a genuine association between the attributes and the class label. If we delve more deeply into the model and analyze the performance of individual attributes, it becomes clear that a model containing only one of the nine attributes can account for nearly all of the elevated accuracy associated with the model. That attribute — *DirectorLastName* — is unusual because it contains such a large number of values. Indeed, individual values sometimes refer to only a single object in the data (e.g., Alfred Hitchcock).

This information about the naïve Bayes model raises an important question — What knowledge does this simple model express?

One way of characterizing the observed behavior is that it represents a relational version of the "small disjuncts" problem (Holte, Acker, & Porter 1989). This problem arises in non-relational learning when overfitted models parse the instance space into sets ("disjuncts") containing only a few data instances. For example, when a decision tree is pathologically large, its leaf nodes can apply to only a single instance in the training set. Such models perform as lookup tables that map single instances to class labels. They achieve very high accuracy on training sets, but they generalize poorly to test sets, when those test sets are statistically independent of the training set.

In the relational version of the small disjuncts problem, models use relational attributes to parse the space of *peripheral* objects into sets so small that they can uniquely identify one peripheral object (e.g., a single Director). If that object is linked to many core objects of a single class (e.g., *genre = comedy*), then a model that uniquely identifies that object can perform well on the training data. If that peripheral object appears in both training and test data, then the model can also perform well on test data.

These conditions certainly hold in the movie data. Single directors are often linked to multiple movies, directors often direct movies in a small number of movie genres, and each value of *DirectorLastName* often identifies no more than one director. For example, Table 1 shows the frequency distributions of *genre* for directors linked to more than 25 movies. The distributions clearly differ by director. Ingmar Bergman has directed 28 movies in the data set, and 26 were dramas. Ninety-eight percent of all films directed by Robert Stevens are in the genre of suspense.

We intentionally included *DirectorLastName* in this experiment to evaluate the effect of small disjuncts when classifying movies, although we could have tested the effect by learning classification trees or rules which can use logical combinations of several attributes having fewer values than *DirectorLastName*. In either case, an overfitted model can uniquely identify directors, and then exploit that information when a director appears in both the training and test sets.

However, viewing the observed behavior of the movie classifier as merely the result of the *small disjuncts* problem ignores an important feature of the movie data. The attribute *genre* exhibits *relational autocorrelation*. That is, when objects x and y are "closely related", then:

¹ We avoided the more traditional approach of 10-fold cross-validation for reasons explained in section 5.

$$P(g(x) | g(y)) \neq P(g(x))$$

where $g(x)$ represents the value of some attribute on x . In the case of *genre* in the movie data, “closely related” refers to the linkage of two movies through a common director.

Relational autocorrelation is similar to temporal autocorrelation that arises in many econometric problems and other time-series prediction tasks, and it is similar to spatial autocorrelation which arises in statistical models in geography and computer vision.

Relational autocorrelation is easy to demonstrate in the movie data. An expanded version of the contingency table shown in Table 1 can be constructed and analyzed with traditional statistical significance tests. The relationship between genre and director can be tested under the null hypothesis of independence. The relationship is significant at less than the 0.1% level.

Name	ActAdv	Comd	Dram	HistDoc	Other	Rom	ScFiF	Susp	Total
A. Konigsberg	0	15	3	0	0	7	0	0	25
A. Hiller	1	7	4	1	0	1	0	18	32
E. Lubitsch	0	10	9	0	4	9	0	0	32
F. Strayer	0	28	0	0	0	0	1	0	29
G. Thomas	0	30	0	0	0	0	0	0	30
G. Cukor	0	6	12	1	4	4	1	1	29
H. Daugherty	0	0	0	0	0	0	0	28	28
I. Bergman	0	0	26	0	0	1	1	0	28
J. Huston	6	3	8	2	0	2	0	4	25
M. Kertesz	8	1	4	3	5	5	1	3	30
P. von Henreid	0	0	1	0	0	2	0	29	32
R. Altman	1	6	7	3	1	0	2	5	25
R. Stevens	0	0	0	0	0	0	1	49	50
R. Stevenson	2	3	3	1	0	0	9	7	25
S. O’Fearn	12	3	8	3	0	1	0	0	27
A. Hitchcock	2	0	7	0	0	3	0	65	77

Table 1: Frequency distributions of movie genre for all directors linked to 25 or more movies. The distributions differ significantly by director ($p < 0.001$).

Autocorrelation is a very common feature of relational data sets. For example, we have observed relational autocorrelation among movie genre with respect to other objects (e.g., studios). We have also observed autocorrelation in a data set on the industry structure of the chemical and banking industries formed from SEC records. Objects in that data set represent companies, officers, directors, owners, as well as ancillary companies such as auditing firms, legal firms, and stock transfer agents. In the SEC data, industry sector exhibits autocorrelation with respect to officers and directors, indicating that persons who sit on multiple boards of directors tend to stay within either banking or chemicals. Autocorrelation was not observed with respect to auditing firms, indicating that the relatively small number of auditing firms tend to target companies in all industries equally.

Relational autocorrelation can be exploited to boost the predictive accuracy of predictive models. For example, our own work has explored *iterative classification*, a technique that builds models based on relational attributes that aggregate predicted class labels of related objects (Neville and Jensen 2000). Iterative classification can be used to identify the predominant genre of a director from information in the test set alone, and then use that information to improve the accuracy of models of a movie’s genre. Work by Slattery and Mitchell (2000) explores other methods of explicitly accounting for shared objects in classification of relational data sets. These methods allow classifiers to represent the valid statistical regularities in relational data (e.g., that directors tend to stay within a genre),

rather than erroneously concluding that a particular last name (or a unique collection of other attributes) predisposes a director to a genre.

4. Sampling to Detect Autocorrelation

Given that autocorrelation is an important issue in applying data mining algorithms to relational data, our evaluation techniques should be able to test for the existence of relational autocorrelation, and to remove the effects of such autocorrelation when desired. This would allow investigators to clearly understand what effects are responsible for the performance of a given model, and thus understand what features are necessary in new data for models to perform well. For example, if a model to predict movie *genre* relies on *DirectorLastName* and relational autocorrelation with respect to director, then a model learned from movies in the 1950s is unlikely to work well on data from the 1990s, because few directors worked in both decades. In contrast, the model could be expected to perform well on randomly selected data from the early 1960s.

One approach to detecting relational autocorrelation is to test all possible subsets of objects and links that could form connections among the objects being classified. This approach would be a natural one for investigators familiar with data mining, but it has several drawbacks. First, such an approach could be prohibitive when data contain hundreds of possible attributes and many possible subsets of objects through which relational autocorrelation could occur. More importantly, it only allows us to determine that relational autocorrelation exists, but does not allow us to determine the effect on the accuracy of a predictive model nor how the model would perform in the absence of autocorrelation.

The remainder of the paper describes and evaluates an alternative approach — sampling relational data in ways that allow some types of relational autocorrelation and eliminate other types. These techniques allow investigators to train and test of models with the effects of particular types of relational autocorrelation entirely removed, and thus check for specific types of autocorrelation.

Sampling Relational Data

Nearly every data mining algorithm is applied to samples of data drawn from a much larger population of possible instances. Even after one such sample exists, a sampling algorithm is often applied again to further divide the sample into training and test sets for algorithm evaluation, to conduct n-fold cross-validation internal to algorithms, to examine algorithm performance as sample size is varied (Oates and Jensen 1998), or to improve the computational efficiency of a learning algorithm (Srinivasan 1999).

Most sampling in machine learning and data mining assumes that instances are independent and identically distributed (iid). A few areas of data mining explicitly use non-random sampling, but either this work is relatively unusual (e.g., active learning and heterogeneous uncertainty sampling (Lewis and Catlett 1994)) or highly specific (e.g., time series analysis). Most general sampling techniques assume that drawing one instance from a population should not affect the probability of inserting any other instance into that sample.

In contrast, this paper examines situations where instances are not iid, and our samples are created with the ultimate goal of constructing and evaluating predictive models. Methods for sampling relational data are not well understood. In the relatively few cases where researchers have considered special methods for sampling relational data for machine learning and data mining, they have often relied on special characteristics of the data. For example, some researchers have exploited the presence of naturally occurring,

disconnected subsets in the population, such as multiple websites without connections among the sites (e.g., Craven et al. 1998). We wish to evaluate classification models that operate over completely connected graphs. There is also a small body of literature on sampling in relational databases (e.g., Lipton et al. 1993), but this work is intended to aid query optimization while our interest is to facilitate evaluation of predictive models.

A sampling algorithm for relational data assigns core and peripheral objects to samples. Recall that core objects are the objects being classified and peripheral objects are all objects needed to calculate relational attributes for a core object. One core object and all its peripheral objects form a *subgraph*. When a sample completely contains a subgraph, the subgraph j of sample k is denoted S_{jk} (the sample subscript is omitted when not necessary for clarity). An object i is denoted O_i . The predicate $in(O_i, S_j)$ indicates that object i is contained in subgraph j . The predicate $core(O_i, S_j)$ indicates that object i is the core object of subgraph j , and $periphery(O_i, S_j)$ indicates that object i is in the periphery of subgraph j . For any given subgraph, the core and periphery are mutually exclusive, and every object in a given subgraph is either in the core or the periphery. Thus, for all i, j :

$$\begin{aligned} core(O_i, S_j) &\Leftrightarrow \neg periphery(O_i, S_j) \\ in(O_i, S_j) &\rightarrow core(O_i, S_j) \vee periphery(O_i, S_j) \end{aligned}$$

Note that an individual object can appear as the core of more than one subgraph, as a peripheral object of more than one subgraph, or as the core of some subgraphs and as a peripheral object of other subgraphs.

An algorithm may, or may not, use knowledge about the subgraph membership of objects when constructing the samples. Below, we analyze one technique that ignores subgraph membership and three others that use subgraph membership to decide which objects to add to a given sample. We evaluate each in terms of how the estimate accuracy in the face of relational autocorrelation. We also evaluate their *sampling efficiency* — the fraction of all instances that can be placed into one of several disjoint samples. In random sampling of iid data, sampling efficiency is always 100% (all instances can be placed into one of the samples). In non-random sampling algorithms, sampling efficiency can be lower than 100%. For example, in a population of 50 women and 50 men, a stratified sampling technique that draws a sample containing 66% women (e.g., 50 women and 25 men) would have only 75% sampling efficiency. As we show below, the efficiency of subgraph sampling can also be lower than 100%.

Object Sampling

The simplest technique for sampling relational data is *object sampling* — each object is placed in a sample selected at random, without regard to other objects already in that sample. If two objects are both placed in a given sample, then the links that connect those two objects in the population are also included in the sample. Object sampling is 100% efficient, because all objects are placed in a sample, and it also produces independent samples.

However, object sampling nearly always introduces both bias and variance into the values of relational attributes, and the magnitude of these effects is inversely proportional to the size of samples. These errors arise because object sampling causes subgraphs to be fragmented across multiple samples. Under object sampling, if a given core object is assigned to a particular sample, it is extremely unlikely that all its peripheral objects will be assigned to the same sample. For a given core object, the probability that all peripheral objects will be assigned to the same sample is p^n where p is the percentage of all objects assigned to the core's sample and n is the total number of peripheral objects for the given core object. Even in extremely favorable conditions (e.g., 50/50 sampling and 5 periph-

eral objects), the probability that a given core object will have a complete periphery is very small (0.03125). In less favorable conditions (10-fold cross-validation and 25 peripheral objects), the probability of a complete periphery is effectively zero (10^{-25}).

Splitting subgraphs across samples causes *fragmentation effects* in relational data. Some of these effects arise because several techniques for ORN learning rely on relational attributes that apply aggregation functions to attribute values of peripheral objects. For example, relational attributes for movies could include the maximum age of any actor in the movie, the total number of actors, and the mean number of movies made by actors in the movie. Popular aggregation functions include *sum*, *maximum*, *minimum*, *count*, *mean*, *median*, *mode*, and *standard deviation*. These types of aggregation functions are used extensively in probabilistic relational models and in our own learning algorithms for PROXIMITY.

Subgraph fragmentation causes bias and increased variance in aggregate attributes. With some aggregate attributes (those using functions such as *sum*, *maximum*, *minimum*, and *count*), fragmented subgraphs leads to statistical bias in attribute estimates. In each case, removing objects from a subgraph can only cause the estimated value of the aggregate attribute to move in a single direction, thus leading to a biased estimate of the true value obtained from the complete subgraph. For example, the aggregate attribute *maximum-actor-age* for a given movie can only remain the same or decrease as peripheral objects (e.g., actors) are removed from the sample containing the core movie. In other aggregate attributes (those using functions such as *mean*, *median*, *mode*, and *standard deviation*), a fragmented subgraph leads to increased variance in attribute estimates. In each of these cases, removing objects from a subgraph decreases the number of data points available to estimate these aggregate values, and thus fragmentation increases the variance (conventionally called the *standard error*) of that estimate.

Another fragmentation effect arises because some techniques for ORN learning rely on path attributes that measure the distance from the core object to specific types of peripheral objects. For example, a path attribute might indicate the minimum path length from a core movie to an Oscar-winning movie, traversing the graph through actors and directors. Any movie starring Clark Gable would have a maximum path length of 1, because of his role in the Oscar-winning movie “Gone with the Wind.” An Oscar-winning movie itself would have a path length of zero. Path attributes are not yet extensively used in relational learning, but are common in analyses of social networks (Wasserman and Faust 1994), scientific citations, and the World Wide Web (Watts 1999). Path attributes have also been popularized by concepts such as Erdős number (the number of traversals through coauthored papers necessary to reach from a given person to the mathematician Paul Erdős) and the game “Six Degrees of Kevin Bacon” (where participants attempt to link a given actor to Kevin Bacon by traversals through movies with multiple actors).

Subgraph fragmentation causes positive bias in path attributes. Subgraph fragmentation removes objects (and thus links) from subgraphs, and thus can increase the estimated values path length attributes. Removing a peripheral object either leaves path length unchanged or increases it, and thus subgraph fragmentation can lead to positive bias in estimates of the value of path attributes. In the worst case, subgraph fragmentation can make the estimated path length essentially infinite by making a destination object entirely unreachable.

How do these types of bias and increased variance affect the accuracy of learned models? Bias will cause systematic errors in parameter estimates internal to models (e.g., the split points of decision trees or the probability density estimates in Bayesian nets and naïve Bayesian classifiers). The effect of attribute bias will be particularly pronounced when training and test sets differ in size, because attribute values in each will be biased to dif-

fering degrees. For example, object sampling for 10-fold cross-validation would produce training sets with (on average) 90% complete subgraphs, but test sets with subgraphs that are only 10% complete. Increased variance due to subgraph fragmentation will act as a type of attribute noise, which can substantially reduce the accuracy of models (Quinlan 1986). Together, these fragmentation effects indicate that object sampling will produce unrepresentative samples and that alternatives to objects sampling should be strongly considered.

Subgraph Sampling

Subgraph sampling guarantees that entire subgraphs appear within specific samples. Recall that we defined subgraphs as a single core object and all peripheral objects necessary to construct a given set of relational attributes for the core. Sampling entire subgraphs preserves the association between each core object and all the peripheral objects necessary for accurate calculation of attributes.

Table 2 lists a generic algorithm for subgraph sampling. The algorithm first assigns subgraphs to prospective samples, and then incrementally converts prospective assignments to permanent assignments only if the subgraphs are separated from subgraphs already assigned to samples. Several possible definitions of separation are discussed in the next section. This conversion from prospective to permanent samples is done m items at a time, where the value of m varies by sample to allow creation of samples of unequal sizes.

Table 2: *Algorithm for Subgraph Sampling*

```

Identify all subgraphs  $S_i$ 

Create prospective samples  $P_1, P_2, \dots, P_n$ , and randomly assign each
subgraph to one sample

Create final samples  $F_1, F_2, \dots, F_n$ , and initialize each to the empty
set.

Loop until at least on  $P_j$  is empty
  For each prospective sample  $P_j$ 
    Do until  $m_j$  subgraphs moved to  $F_j$ 
    or  $P_j$  empty
      Select subgraph  $S_{ij}$  from  $P_j$ 

      If separate( $S_{ij}, S_{\neq k}$ ) for all  $j \neq k$ ,
      then move  $S_{ij}$  from  $P_j$  to  $F_j$ ,
      else discard  $S_{ij}$  from  $P_k$ .

Return final samples  $F_1, F_2, \dots, F_n$ 

```

Two features of the algorithm are worthy of special note. First, the algorithm enforces separation between subgraphs in different samples to promote statistical independence. As sampling progresses, subgraphs are discarded if they are not separate from subgraphs already assigned to another sample. This feature of the algorithm is intended to prevent specific types of relational autocorrelation among instances in different samples. Three different criteria for subgraph separation are discussed below.

A second feature of the algorithm — the random assignment of subgraphs to “prospective” samples — also promotes statistical independence. The algorithm either makes a prospective assignment permanent, or discards the subgraph. An alternative algorithm would search for an assignment of permanent labels to subgraphs that the maximizes

number of subgraphs assigned to each sample. This approach would maximize sampling efficiency (one of the three goals of sampling outlined above).

However, this approach to maximizing sampling efficiency reduces the statistical independence of multiple samples. Consider how such an “optimized” algorithm would behave when confronted with a data set consisting of two disjoint (or nearly disjoint) sets of relational data. Sampling efficiency could be optimized by filling one sample entirely with objects from one disjoint set, and filling another sample with objects from the other set. If the statistical characteristics of one of the disjoint sets did not mirror the characteristics of each other, then accuracy estimates of learned models will be biased downward.

We encountered precisely this problem when sampling relational data about the industrial structure of the banking and chemical industries (Neville and Jensen 2000). Our task was to create classifiers that predicted the industrial sector of a company (*banking* or *chemical*) based on the company’s intrinsic and relational attributes. When we created training and test samples for this task, we attempted to optimize sampling efficiency, but we encountered great difficulty obtaining samples with roughly equivalent distributions of the class label. Several of our relational attributes involved companies linked through officers and directors. Officers and directors often link to multiple companies (e.g., an officer of one company and a director of two others), but they almost always stay within a single industry (their area of expertise). As a result, attempting to optimize sampling efficiency harmed statistical independence because it tended to produce training and test samples drawn from different industrial sectors. For this reason, the subgraph sampling algorithm in Table 1 assigns prospective samples randomly and then either confirms that assignment or discards the subgraph. In addition, we stop assigning subgraphs to samples as soon as one of the set of prospective classes has been exhausted.

Subgraph sampling is similar to *snowball sampling*, a sampling technique used in social network analysis (Goodman 1961). In snowball sampling, a single seed object is inserted into a sample, and then all its neighbors are inserted, and then all those objects’ neighbors, and so on until a desired sample size (or a specified network boundary) is reached. Snowball sampling was developed for use with *homogeneous* relational data where all objects have the same type (e.g., data where all objects are persons and all links represent acquaintanceship or telecommunications data where all objects are phone numbers and all links are calls). In ORN learning applications, however, objects can have heterogeneous type. Simple snowball sampling could still fragment subgraphs at the boundaries of a sample. Subgraph sampling avoids this fragmentation by accounting for the different roles of core and peripheral objects in attribute calculation.

Subgraph sampling also resembles techniques that construct samples from a small number of completely disconnected graphs. For example, some experiments with WEBKB train classification models on pages completely contained within a single website, and then test those models on pages from another website with no links to the training set. This approach exploits a feature of some websites — heavy internal linkage but few external links. Similarly, some work in ILP constructs samples from sets of completely disconnected graphs (e.g., individual molecules or English sentences). This approach are possible only when the domain provides extremely strong natural divisions in the graphs, and this approach is only advisable where the same underlying process generated each graph. In contrast, subgraph sampling can be applied to data without natural divisions. Where they exist, subgraph sampling will exploit some types of natural divisions. Where they do not exist, logical divisions among subgraphs can be created that preserve the ability to accurately calculate relational attributes and yet preserve the statistical independence among samples.

Determining Subgraph Separation

The algorithm for subgraph sampling (Table 2) depends on the predicate $separate(S_i, S_j)$ which indicates whether two subgraphs consist of disjoint sets of objects. We differentiate among three criteria for determining subgraph separation. The criteria are differentiated by the degree to which elements of the two subgraphs are separated. Figure 2 shows the criteria schematically. Informal descriptions and formal definitions are given below.

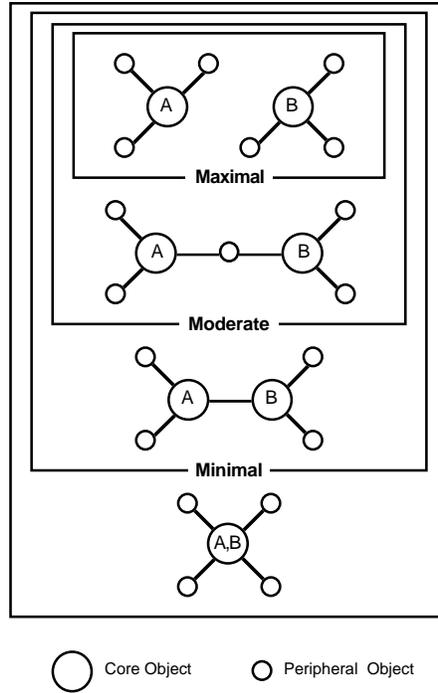


Figure 2: Three types of subgraph separation. Subgraphs of core objects A and B are in different samples.

Minimal separation specifies only that subgraphs in different samples have distinct cores, but allows peripheral objects of subgraphs in one sample to appear in the periphery or core of subgraphs in another sample.¹ That is, for all i, m, n, x, y , where $x \neq y$,

$$core(O_i, S_{m_x}) \rightarrow \neg core(O_i, S_{n_y})$$

Minimal separation specifies that a single object cannot serve as the core object of a subgraph in more than one sample. This criterion mirrors the constraint on traditional iid sampling, where an instance cannot appear in more than one sample. However, it allows both core and peripheral objects in one sample to appear as peripheral objects in another sample. *Minimal separation* was used the experiment discussed in Section 2, and thus results obtained using this sampling technique can be affected by relational autocorrelation. That said, *minimal separation* provides 100% sampling efficiency, because all possible core objects can appear in a sample.

¹ For ease of explanation, we assume that a single object never serves as the core object of two or more subgraphs within a single sample.

Moderate separation adds the condition that the core object of each subgraph is distinct from all peripheral objects in subgraphs of other samples, but still allows peripheral objects of subgraphs in one sample to appear as peripheral objects of subgraphs in another sample. That is, for all i, m, n, x, y , where $x \neq y$,

$$\begin{aligned} \text{core}(O_i, S_{mx}) &\rightarrow \neg \text{core}(O_i, S_{ny}) \\ \text{core}(O_i, S_{mx}) &\rightarrow \neg \text{periphery}(O_i, S_{ny}) \end{aligned}$$

Moderate separation can provide 100% sampling efficiency if no links directly connect core objects, or if such links exist, but are not used to create relational attributes. For example, the movie data contain almost no direct links between movies (*remake-of* links are the only exception). Thus, judicious creation of relational attributes could allow 100% sampling efficiency even with moderate separation.

Maximal separation adds the condition that the peripheral objects of each subgraph are distinct from the peripheral objects of subgraphs in other samples. Thus, in maximal separation, objects of subgraphs in one sample never appear as objects of subgraphs in another sample. That is, for all i, m, n, x, y , where $x \neq y$,

$$\begin{aligned} \text{core}(O_i, S_{mx}) &\rightarrow \neg \text{core}(O_i, S_{ny}) \\ \text{core}(O_i, S_{mx}) &\rightarrow \neg \text{periphery}(O_i, S_{ny}) \\ \text{periphery}(O_i, S_{mx}) &\rightarrow \neg \text{periphery}(O_i, S_{ny}) \end{aligned}$$

or, more simply,

$$\text{in}(O_i, S_{mx}) \rightarrow \neg \text{in}(O_i, S_{ny})$$

In contrast to minimal and moderate separation, there is no sharing of objects among samples and thus there is no opportunity for “information leakage” between the training and test sets. However, maximal separation can exact a heavy toll in terms of sampling efficiency. For populations with even a small number of highly connected objects that are used in relational attributes, this approach can become untenable. For example, the movie data contain a small number of studios that link to hundreds of movies. We explore this problem in more detail in the experiments below.

5. Evaluation

We conducted experiments using the movie data and subgraphs of radius $r=1$ (movies and all objects directly linked to them). Using these subgraphs, samples were constructed for each of the three different separation criteria.

Sampling Efficiency

To evaluate the sampling efficiency associated with different separation criteria, we conducted experiments using the movie data. We selected 8,212 movies from the top 20 genres, and created subgraphs of radius $r=1$ (movies and all objects directly linked to them) and $r=2$ (all $r=1$ objects and objects directly linked to them). For each set of subgraphs, we applied subgraph sampling to construct equal-sized training and test sets using each of the three separation criteria.

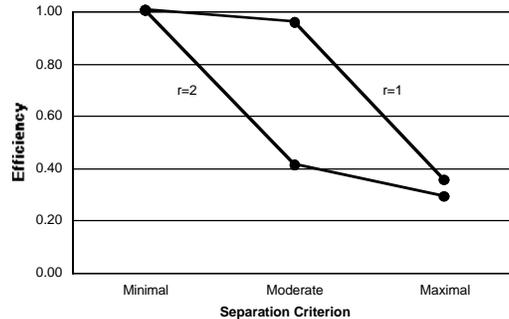


Figure 3: *Sampling efficiency using subgraph sampling with different separation criteria and different-sized subgraphs.*

The sampling efficiency in each of the six cases is shown in Figure 3. Minimal separation is perfectly efficient for the movie data, because no two subgraphs have the same core object. Moderate separation produces only a slight drop in efficiency for $r=1$ because two movies that are remakes cannot appear in different samples. As expected, moderate separation for $r=2$ and maximal separation for $r=1$ and $r=2$ produce far larger drops in efficiency.

These results show the best case of sampling efficiency, because subgraphs were placed into only two samples. In cases where a larger number of independent samples are needed (e.g., 10-fold cross-validation), sampling efficiency would be far lower because each subgraph in any one sample would have to be separate from a much larger percentage of all other subgraphs (e.g., 90% in 10-fold cross-validation, rather than 50% in 50/50 sampling).

Relational Autocorrelation

As outlined in the example in Section 2, we constructed classifiers using *DirectorLastName* as the sole attribute. For each pair of training and test sets, we constructed classifiers on the training set and estimated their accuracy on the test set. Figure 4 shows the results for two types of class label. Each point is an average of five trials. Standard deviations are less than 0.01 for minimal and moderate and less than 0.03 for maximal. Accuracy for the best default classifier (*genre = drama*) is 24%. The default accuracy is shown in the figure with a dotted line.

The upper line shows the accuracy of the classifier on the actual class label. Accuracy on test sets generated using *minimal* and *moderate* separation is approximately 40%, but accuracy drops to default when *maximal* separation is used. In samples generated with maximal separation, directors never occur in both training and test sets, and thus the model cannot take advantage of relational autocorrelation.

The lower line in Figure 4 shows the accuracy of the classifier on versions of the data sets with randomized class labels. The class labels for these sets were artificially generated by randomly sampling from the probability distribution of the actual class label. Thus, the randomized class labels are independent of any attribute (including *DirectorLastName*), and will not exhibit relational autocorrelation. In this case, the classifiers have significantly reduced accuracy when evaluated on samples generated using *minimal* and *moderate* separation.

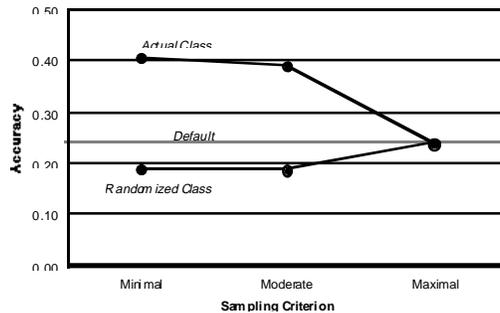


Figure 4: Estimates of test set accuracy of classifiers using subgraph sampling with different separation criteria. All models use the *DirectorLastName* attribute. Results for predicting the actual class are elevated when samples are generated with Minimal and Moderate separation. Results for predicting a randomized class label are depressed due to overfitting.

This effect is due to traditional overfitting. The naïve Bayes classifier estimates $p(\text{class}|\text{DirectorLastName})$, and these estimates are often based on very few values. This was not a substantial problem in the non-randomized data, because class distributions were generally highly skewed toward one value. In the randomized case, however, distributions are more uniform. As a result, the classifier sometimes selects the non-default class as the most probable class, thus reducing accuracy. When samples are generated using *maximal* separation, however, the value of *DirectorLastName* in the test set are almost always missing from the model, and the classifier then falls back on its default estimate of the probability distribution over the class labels, and selects the most probable class.

7. Discussion and Future Work

Relational autocorrelation is an intriguing and potentially very useful characteristic of relational data. We are exploring how it can be explicitly incorporated into probabilistic models, learned directly from data, and effectively evaluated. In addition, we are investigating how relational autocorrelation may affect evaluations of existing techniques for learning from relational data. Given that it is such a common feature of relational data sets, relational learning techniques that are prone to overfitting could inadvertently mistake autocorrelation for other types of statistical associations.

Acknowledgments

The authors thank Matt Cornell and Hannah Blau for their work on components of PROXIMITY. This research is supported by DARPA/AFOSR under contract No. F30602-00-2-0597, DARPA/AFRL under contract No. F30602-99-C-0061, and an NSF Graduate Fellowship. The movie data were provided by the UCI KDD Archive. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright notation hereon. The views and conclusions contained in this paper are solely those of the authors.

References

Craven, M., DiPasquo, D., Freitag, D., McCallum, A., Mitchell, T., Nigam, K., & Slatery, S. (1998). Learning to extract symbolic knowledge from the World Wide Web.

Proceedings of the Fifteenth National Conference on Artificial Intelligence (pp. 509-516). Menlo Park: AAAI Press.

Friedman, N., Getoor, L., Koller, D., & Pfeffer, A. (1999). Learning probabilistic relational models. *Proceedings of the 1999 International Joint Conference on Artificial Intelligence* (pp. 1300-1307). San Francisco: Morgan Kaufmann.

Goodman, L. (1961). Snowball sampling. *Annals of Mathematical Statistics*, 32, 148-170.

Holte, R., Acker, L., & Porter, B. (1989). Concept learning and the accuracy of small disjuncts. Proceedings of the 11th International Joint Conference on Artificial Intelligence (pp. 813-818). Detroit: Morgan Kaufmann.

Lewis, D. & Catlett, J. (1994). Heterogeneous uncertainty sampling for supervised learning. *Proceedings of the 11th International Conference on Machine Learning* (pp. 148-156). San Francisco: Morgan Kaufmann.

Lipton, R., Naughton, J., Schneider, D., & Seshadri, S. (1993). Efficient sampling strategies for relational database operations. *Theoretical Computer Science*, 116, 195-226.

Muggleton, S. (1992). *Inductive Logic Programming*. San Diego: Academic Press.

Neville, J. & Jensen, D. (2000). Iterative classification in relational data. *Papers from the AAAI Workshop on Learning Statistical Models from Relational Data* (pp. 42-49). Menlo Park: AAAI Press.

Oates, T. & Jensen, D. (1997). The effects of training set size on decision tree complexity. Proceedings of The 14th International Conference on Machine Learning (pp. 254-262). San Francisco: Morgan Kaufmann.

Slattery, S. & Mitchell, T. (2000). Discovering test set regularities in relational domains. *Proceedings of the 17th International Conference on Machine Learning*. San Francisco: Morgan Kaufmann.

Srinivasan, A. (1999). A study of two sampling methods for analysing large datasets with ILP. *Data Mining and Knowledge Discovery*, 3, 95-123.

Quinlan, J. R. (1986). The effect of noise on concept learning. In Michalski, R. S., Carbonell, J., and Mitchell, T. M., (eds.), *Machine learning, Vol. II*. Palo Alto: Tioga Press.

Wasserman, S. & Faust, K. (1994). *Social Network Analysis*. Cambridge: Cambridge University Press.

Watts, D. (1999). *Small Worlds*. Princeton, NJ: Princeton University Press.