Autocorrelation and Linkage Cause Bias in Evaluation of Relational Learners

David Jensen and Jennifer Neville

Department of Computer Science
140 Governors Drive
University of Massachusetts, Amherst
Amherst, MA 01003
{jensen|jneville}@cs.umass.edu

Two common characteristics of relational data sets — concentrated linkage and relational auto-correlation — can cause traditional methods of evaluation to greatly overestimate the accuracy of induced models on test sets. We identify these characteristics, define quantitative measures of their severity, and explain how they produce this bias. We show how linkage and autocorrelation affect estimates of model accuracy by applying FOIL to synthetic data and to data drawn from the Internet Movie Database. We show how a modified sampling procedure can eliminate the bias.

Introduction

Accurate evaluation of learning algorithms is central to successful research in relational learning. The most common method for evaluating a learning algorithm is to partition a given data sample into training and test sets, construct a model using the training set, and evaluate the accuracy of that model on the test set. Separate training and test sets are used because of a widely observed bias when the accuracy of models is assessed on the original training set (Jensen & Cohen 2000).

In this paper, we show how dependence among the values of a class label in relational data can cause strong biases in the estimated accuracy of learned models when accuracy is estimated in this conventional way. In this section, we give a simple example of how estimated accuracy can be biased. In later sections, we define quantitative measures of concentrated linkage (L) and relational autocorrelation (C'), two common characteristics of relational data sets. We show how high values of L and C' cause statistical dependence between training and test sets, and we show how this dependence leads to bias in test set accuracy. In general, current techniques for evaluating relational learning algorithms do not account for this bias, although we discuss some special classes of relational data sets that are immune to this effect. We present a new family of sampling algorithms that can be applied to any relational data set, and show how it eliminates the bias.

These results indicate that accurate evaluation of relational learning algorithms will often require specialized evaluation procedures. The results also indicate that

D. Jensen and J. Neville (2002). Autocorrelation and linkage cause bias in evaluation of relational learners. To appear in *Proceedings of The Twelfth International Conference on Inductive Logic Programming (ILP 2002)*. Springer-Verlag.

additional attention should be paid to identifying and using relational autocorrelation to improve the predictive accuracy of relational models. This paper is part of a larger study of the effects of linkage and autocorrelation on relational learning. A related paper (Jensen and Neville 2002) shows how linkage and autocorrelation affect feature selection in relational learning.

Statistical Analysis of Relational Data

Recent research in relational learning has focused on learning statistical models, including work on stochastic logic programming (Muggleton 2000), probabilistic relational models (Getoor et al. 1999), and relational Bayesian classifiers (Flach and Lachiche 1999). However, with the greater expressive power of relational representations come new statistical challenges. Much of the work on relational learning diverges sharply from traditional learning algorithms that assume data instances are statistically independent. The assumption of independence is among the most enduring and deeply buried assumptions of machine learning methods, but this assumption is contradicted by many relational data sets.

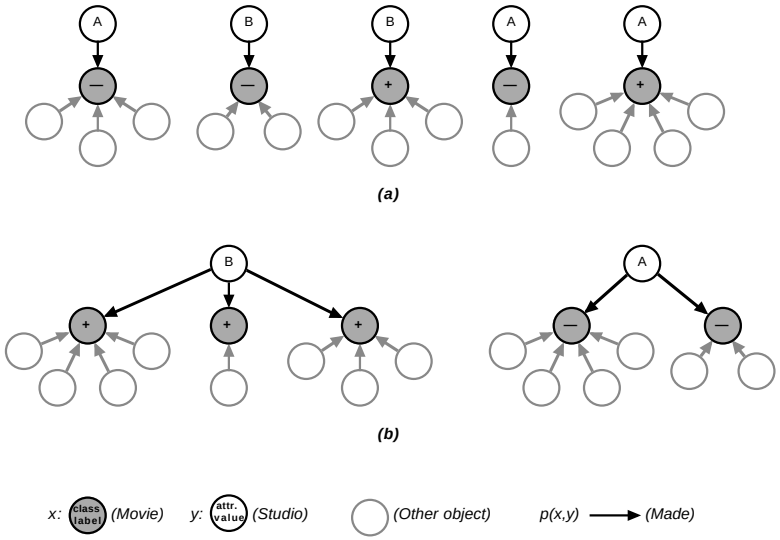

Fig. 1. Example relational data sets (a) five independent instances and (b) five dependent instances.

For example, consider the two simple relational data sets shown in Figure 1. In each set, instances for learning consist of subgraphs containing a unique object *x*, an object *y*, and one or more other objects. Objects *x* contain a class label and objects *y* contain an attribute that will be used to predict the class label of *x*. Figure 1a shows a data set where objects *x* and *y* have a one-to-one relationship and where the class labels on instances are independent. Figure 1b shows instances where objects *x* and *y* have a many-to-one relationship and where the class labels are dependent.

We spend the majority of this paper considering data sets similar in structure to Figure 1b, where each subgraph consists of multiple relations and each relation may produce statistical dependencies among the instances. For simplicity, all relations in Figure 1 are binary, but the statistical effects we investigate affect a wider range of tasks and data representations.

Simple Random Partitioning

Perhaps the most widely used evaluation technique in machine learning and data mining is the partioning of a data sample into training and test sets. Most sampling in machine learning and data mining assumes that instances are independent. In contrast, this paper examines situations where instances are not independent. 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.

The most common sampling technique — *simple random partitioning* — has been taken from propositional settings and adapted for use in relational sets such as those in Figure 1. We define simple random partitioning with respect to two sets of objects X and Y and a set of paths P such that the relation $p(x,y)$ holds. We presume that objects X contain class labels and that objects X and Y both contain attributes relevant to classifying objects X. Paths are composed of k links and $k-1$ intervening objects, where $k \geq 1$. Each path represents a series of relations linking an object in X to an object in Y. For example consider the path linking two movies, m_1 and m_2, made by the same studio. The path is formed from two $Made$ links, $Made(m_1,s_1)$ and $Made(m_2,s_1)$. For convenience we treat all links as undirected in order to refer to meaningful sequences of relationships as paths. We assume that paths in P are unique with respect to a given (x,y) pair; if two or more paths between x and y exist in the data, they are collapsed to a single element of P.

> **Definition:** *Simple random partitioning* divides a sample S of relational data into two subsamples S_A and S_B. The subsamples are constructed by drawing objects X from S without replacement and without reference to paths P and objects Y in S. When an individual object $x \in X$ is placed into a subsample, some set of objects $\{y_1, y_2, ...y_n\}$, such that $p(x,y_i)$, are also placed into the subsample if those objects are not already present. The object sets X_A and X_B are mutually exclusive and collectively exhaustive of objects X in S, but other objects Y may appear in both S_A and S_B.

Such a techique for creating training and test sets seems a logical extension of techniques for propositional data. However, it leaves open the possibility that a subset

of objects *Y* may fall into both the training and test set, creating some type of dependence between the training and test sets.

An Example of Test Set Bias

Given that instances in relational data may share some objects, and thus not be independent, we should examine how such relational structure could affect accuracy estimates made using training and test sets. Below we show how relational structure and dependence among values of the class label can bias estimates of the accuracy of induced models.

We created data sets about movies and analyzed them using FOIL (Quinlan 1990). Specifically, we created and analyzed simple relational data sets whose relational structure was drawn from the Internet Movie Database (www.imdb.com). We gathered a sample of all movies in the database released in the United States between 1996 and 2001 for which we could obtain information on box office receipts. In addition to 1382 movies, the data set also contains objects representing actors, directors, producers, and studios.[1] In all, the data set contains more than 40,000 objects and almost 70,000 links. The data schema is shown in Figure 2.

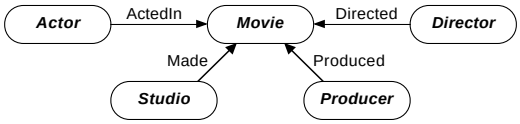

Fig. 2. A general data schema for the movie data sets.

For most of the experiments reported in this paper, we limited analysis to just two classes of objects — movies and studios. This allowed us to greatly reduce the overall size of training and test sets, thus making them feasible to analyze using FOIL. This also focused the experiments on precisely the phenomena we wished to study, as discussed below. Details about the data representation are given in the appendix.

We created a learning task using an attribute on movies — opening-weekend box office receipts. We discretized the attribute so that a positive value indicates a movie with more than $2 million in opening weekend receipts (*prob(+)=0.55*). We call this discretized attribute *receipts* and use it as a binary class label. We also created ten random attributes on studios. The values of these attributes were randomly drawn from a uniform distribution of five values, and thus were independent of the class label. Figure 3 shows the schema with movies, studios, and their attributes.

We used simple random partitioning to create (approximately) equal-sized training and test sets. This divides our sample of 1382 movies in two subsamples, each containing approximately 690 movies and their affiliated studios. Each movie appears in only one sample. Each affiliated studio might appear in one or both subsamples,

[1] Each of the movies is related to one primary studio. For movies with more than one associated studio, we chose the U.S. studio with highest degree to be the primary, for movies without any U.S. studios we chose the studio of highest degree to be the primary.

and would never appear more than once in any one subsample. However, a single studio object can be linked to many movies in a given subsample.

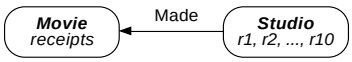

Fig. 3. A simplified data schema, with attributes, for the artificial data sets used for experiments in this section. Attributes denoted *r1* through *r10* are random attributes.

Given these data sets, we evaluated the ability of FOIL to learn useful models in the traditional way. We ran FOIL on the training set and evaluated the accuracy of the resulting models on the test set. Given that attributes on studios were created randomly, the expected error for the models constructed exclusively from those attributes should equal the default error (0.55). Deviations from this error represent a bias, which can be measured by subtracting the measured error $\hat{e}$ from the theoretical error e. Positive bias over many trials indicates that the test set accuracy is systematically lower than the theoretical error.

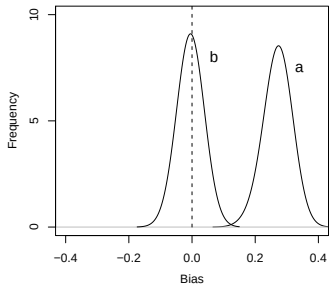

Fig. 4. Distribution of test set bias for FOIL models constructed from random attributes using (a) the actual class labels and (b) randomized class labels. The distributions were smoothed using a bandwidth parameter of 0.4.

Figure 4 shows two distributions of bias estimated from 50 different training and test set partitions. The rightmost distribution (a) results from the experiment described above. The bias is substantially larger than zero, indicating that the measured error of FOIL rules on the test set is much lower than the default. The leftmost distribution (b) results from running the same experiment, except that the values of the class label on movies (*receipts)* are randomly reassigned before each trial. This distribution has almost precisely the expected bias of zero.

These experimental results raise obvious questions: Why does the algorithm appear to learn from random features formed from studios when the actual class label is used, but not when the values of that class label are randomly assigned? What does this result tell us about evaluating relational learners in general?

Linkage, Autocorrelation, and Overfitting

Our analysis indicates that biases like that shown in Figure 4 result from the confluence of three common phenomena in relational learning — *concentrated linkage*, *relational autocorrelation*, and *overfitting*. Concentrated linkage and relational autocorrelation create statistical dependence among instances in different samples, and overfitting exploits that dependence in ways that lead to bias in test set accuracy. We define concentrated linkage and relational autocorrelation formally below. Informally, concentrated linkage occurs when many objects are linked to a common neighbor, and relational autocorrelation occurs when the values of a given attribute are highly uniform among objects that share a common neighbor. Overfitting is familiar to machine learning researchers as the construction of complex models that identify unique characteristics of the training set rather than statistical generalizations present in the population of all data.

Much of the text of this section is drawn from an earlier paper (Jensen & Neville 2002). However, the definitions are so central to understanding the experiments in later sections that we present this material again rather than attempting to summarize.

Concentrated Linkage

We define concentrated linkage $L(X,P,Y)$ with respect to the same conditions as simple random partioning — two sets of objects X and Y and a set of paths P such that $p(x,y)$.

Definition: D_{yX} is the degree of an object y with respect to a set of objects X. That is, the number of $x \in X$ such that $p(x,y) \in P$. For example, D_{yX} might measure, for a given studio y, the number of movies (X) it has made. ■

Definition: *Single linkage* of X with respect to Y occurs in a data set whenever, for all $x \in X$ and $y \in Y$:

$$D_{xY} = 1 \quad \textit{and} \quad D_{yX} \geq 1$$ ■

In these cases, many objects in X (e.g., movies) connect to a single object in Y (e.g., a studio). We use single linkage as an important special case in future discussions.

Definition: The *concentrated linkage $L(x,X,P,Y)$ of an individual object x* (e.g., a movie) that is linked to objects Y (studios) via paths P is:

$$L(x,X,P,Y) = \sum_{\substack{y \ s.t. \\ p(x,y) \in P}} \frac{(D_{yX} - 1)}{D_{yX}} \Big/ D_{xY}^{\,2}$$ ■

the quantity $(D_{yX} - 1)/D_{yX}$ within the summation is zero when the D_{yX} is one, and asymptotically approaches one as degree grows, and thus is a reasonable indicator of $L(x,X,P,Y)$, given single linkage of x with respect to Y. Because x may be linked to multiple nodes in Y, we define the average across all nodes y_i linked to x, and divide by an additional factor of D_{xY} to rate single linkage more highly than multiple linkage.

Definition: The *concentrated linkage L(X,P,Y) of a set of objects X* (e.g., all movies) that are linked to objects *Y* is:

$$L(X,P,Y) = \sum_{x \in X} \frac{L(x,X,P,Y)}{|X|}$$ ■

Given particular types of linkage, *L* can be calculated analytically from the sufficient statistics |*X*| and |*Y*|. For example, in the case of single linkage of *X* with respect to *Y*, *L = (*|*X*|-|*Y*|*)*/|*X*|. For example, the data set shown in Figure 1b exhibits single linkage, so *L(X,P,Y)* = 0.60. Propositional data also display single linkage, and because |*X*|=|*Y*|, *L(X,P,Y) = 0*. Calculations of several types of linkage are shown for the movie data in Table 2.

Table 2: Linkage in the movie data

Linkage Type	Value
L(Movie, Made, Studio)	0.91
L(Movie, Directed, Director)	0.23
L(Movie, Produced, Producer)	0.08
L(Movie, ActedIn, Actor)	0.01

In addition to the movie data, we have encountered many other instances of concentrated linkage. For example, while studying relationships among publicly traded companies in the banking and chemical industries, we found that nearly every company in both industries uses one of only seven different accounting firms. In work on fraud in mobile phone networks, we found that 800 numbers, 900 numbers, and some public numbers (e.g., 911) produced concentrated linkage among phones. Concentrated linkage is also common in other widely accessible relational data sets. For example, many articles in the scientific literature are published in single journals and many basic research articles are cited in single review articles. On the Web, many content pages are linked to single directory pages on sites such as Yahoo and Google.

Correlation and Autocorrelation

We will define relational correlation *C(X,f,P,Y,g)* with respect to two sets of objects *X* and *Y*, two attributes *f* and *g* on objects in *X* and *Y*, respectively, and a set of paths *P* that connect objects in *X* and *Y*.

Definition: *Relational correlation C(X,f,P,Y,g)* is the correlation between all pairs *(f(x),g(y))* where $x \in X$, $y \in Y$ and $p(x,y) \in P$. ■

Given the pairs of values that these elements define, traditional measures such as information gain, chi-square, and Pearson's contingency coefficient can be used to assess the correlation between values of the attributes *f* and *g* on objects connected by paths in *P*. The range of *C* depends on the measure of correlation used.

We can use the definition of relational correlation *C(X,f,P,Y,g)* to define relational *autocorrelation* as the correlation between the same attribute on distinct objects belonging to the same set.

Definition: *Relational autocorrelation C'* is:

$$C'(X,f,P) \equiv C(X,f,P,X,f) \quad \text{where} \quad \forall p(x_i,x_j) \in P \quad x_i \neq x_j \qquad \blacksquare$$

For example, *C'* could be defined with respect to movie objects, the attribute *receipts* on movies, and paths formed by traversing *Made* links that connect the movies to an intervening studio.

If the underlying measure of correlation varies between zero and one, then $C'=1$ indicates that the value of the attribute for a specific node x_i is always equal to all other nodes x_j reachable by a path in P. When $C'=0$, values of $f(X)$ are independent. Table 3 gives estimates of relational autocorrelation for movie receipts, linked through studios, directors, producers, and actors. For a measure of correlation, Table 3 uses Pearson's corrected contingency coefficient (Sachs 1992), a measure that produces an easily interpreted value between zero and one. Autocorrelation is fairly strong for all object types except actors.

In addition to the movie data, we have encountered many other examples of high relational autocorrelation. For example, in our study of publicly traded companies, we found that when persons served as officers or directors of multiple companies, the companies were often in the same industry. Similarly, in biological data on protein interactions we analyzed for the 2001 ACM SIGKDD Cup Competition, the proteins located in the same place in a cell (e.g., mitochondria or cell wall) had highly autocorrelated functions (e.g., transcription or cell growth). Such autocorrelation has been identified in other domains as well. For example, fraud in mobile phone networks has been found to be highly autocorrelated (Cortes et al. 2001). The topics of authoritative web pages are highly autocorrelated when linked through directory pages that serve as "hubs" (Kleinberg 1999). Similarly, the topics of articles in the scientific literature are often highly autocorrelated when linked through review articles.

Table 3: Autocorrelation in the movie data

Autocorrelation Type	Value
C'(Movie,Receipts,Made|Studio|Made)	0.47
C'(Movie,Receipts,Directed|Director|Directed)	0.65
C'(Movie,Receipts,Produced|Producer|Produced)	0.41
C'(Movie,Receipts,ActedIn|Actor|ActedIn)	0.17

Note: The notation *a|x|b* to denote paths with links of type *a* and *b* and intervening objects of type *x*.

We define relational autocorrelation in a similar way to existing definitions of temporal and spatial autocorrelation (see, for example, Cressie 1993). Autocorrelation in these specialized types of relational data has long been recognized as a source of increased variance. However, the more general types of relational data commonly analyzed by relational learning algorithms pose even more severe challenges because the amount of linkage can be far higher than in temporal or spatial data and because that linkage can vary dramatically among objects.

Relational autocorrelation represents an extremely important type of knowledge about relational data, one that is just beginning to be explored and exploited for learning statistical models of relational data (Neville and Jensen 2000; Slattery and

Mitchell 2000). Deterministic models representing the extreme form of relational autocorrelation have been learned for years by ILP systems. By representing and using relational autocorrelation, statistical models can make use of both partially labeled data sets and high-confidence inferences about the class labels of some nodes to increase the confidence with which inferences can be made about nearby nodes.

However, as we show below, relational autocorrelation can also greatly complicate learning of all types of relational models. As we seek to represent and use relational autocorrelation in statistical models of relational data, we will need to adjust for its effects when evaluating more traditional types of features in these models.

Effects of Linkage, Autocorrelation, and Overfitting on Bias

The results reported so far for concentrated linkage and relational autocorrelation provide important clues to the behavior shown in Figure 4. Studios are the objects in the movie data that have the highest combination of concentrated linkage and relational autocorrelation. In this section, we show that, if linkage and autocorrelation are both high for a single type of object, and an algorithm produces overfitted models that use attributes on those objects, then the test set error will be biased.

Dependent Training and Test Sets

Given the definition of simple random partitioning in the introduction, we can examine the statistical dependence of subsamples it produces. We define $prob_{ind}(A,B)$ to be the probability that subsamples A and B are independent.[2] Further, we define $prob_{ind}(A,B|y)$ to be the probability that subsamples A and B are independent with respect to a specific object $y \in Y$, that is where A and B are independent with respect to objects $x_i \in X$ such that $p(x_i,y) \in P$.

Theorem: Given simple random partitioning of a relational data set S with single linkage and $C'=1$:

$$prob_{ind}(A,B) \rightarrow 0 \quad as \quad L \rightarrow 1.$$

Proof: First, consider a sample S composed of independent subgraphs such as those shown in Figure 1a. In such a sample, $L=0$ because no x is linked to more than one y, and $prob_{ind}(A,B)=1$ because no object x can fall into more than one sample and there is no dependence among objects x_1 and x_2 because $L = 0$.

Now consider the effect of increasing D_{yX}, the degree of an object y (e.g., a studio) with respect to objects X (e.g., movies). For notational convenience, $d=D_{yX}$. Increasing d necessarily increases L for a single object x, because for single linkage $L(x,X,P,Y)=(d-1)/d$. For any one object y, the samples A and B are not independent if both contain an object x_i, such that $p(x_i,y)$. Thus:

---

[2] In this paper, we consider the effects of dependence between instances in different subsamples, but not the effects of dependence among objects within the *same* subsample. The latter topic is covered in another recent paper (Jensen and Neville 2002).

$$prob_{ind}(A,B \mid y) = b(0,d,p) + b(d,d,p) \quad (1)$$

where $b(m,n,p)$ denotes the value of the binomial distribution for m successes in n trials, where each trial succeeds with probability p. Here, for example, $b(d,d,p)$ is the probability that a given sample (e.g., A) will contain all of the d movies linked to a given studio y. In the case of simple random partitioning into equal-sized samples, $p=0.5$. For high values of d (many objects x are connected to a single object y), $prob_{ind}(A,B|y)$ approaches zero. For $d=2$, $prob_{ind}(A,B|y)=0.5$. That is, there is only a 50% probability that both objects x connected to y will end up in the same sample. For $d=3$, $prob_{ind}(A,B|y)=0.25$; for $d=10$, $prob_{ind}(A,B|y)=0.002$.

Given that S contains many objects Y, the probability of independence for *all* objects y becomes vanishingly small. Specifically:

$$prob_{ind}(A,B) = \prod_{y} prob_{ind}(A,B \mid y) \quad (2)$$

For even small samples, $prob_{ind}(A,B)$ goes quickly to zero as L increases. For example, given a sample of 50 instances of X, if $d=2$ $(L=0.5)$, then $prob_{ind}(A,B)=3.0x10^{-8}$. For $d=5$ $(L=0.8)$, $prob_{ind}(A,B)=9.1x10^{-13}$. For large samples and L>0, $prob_{ind}(A,B) \approx 0$. ■

Not only is the probability of *any* dependence between A and B high, but the degree of dependence is very likely to be high. For example, the binomial distribution can be used to derive the expected number of objects x_1 in sample A with a matching object x_2 in B such that $p(x_1,y) \in P$ and $p(x_2,y) \in P$ for some $y \in Y$ with degree $d=D_{yX}$. The maximum and expected number of matched pairs of dependent instances is:

$$Max(pairs) = \lfloor d/2 \rfloor \quad (3)$$

$$E(pairs) = \sum_{i=1}^{d} \min(i, d-i) b(i,d,0.5)$$

For example, for $d=5$, the maximum number of pairs is $Max(pairs)=2$ and the expected number is $E(pairs)=1.56$. For $d=10$, $Max(pairs)=5$ and $E(pairs)=3.77$.

How Bias Varies with Autocorrelation, Linkage, and Overfitting

Given dependent training and test sets, it is relatively easy to see how overfitted models can cause bias in test set accuracy. 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 propositional 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 objects *y* into sets so small that they can uniquely identify one such object (e.g., a single studio). If that object is linked to many objects *x* where a single class predominates (e.g., *receipts* = +), then a model that uniquely identifies that object *y* can perform well on the training data. If that *y* also appears in the test data, then the model can perform well on test data.

Indeed, such overfitting is made more likely by high autocorrelation among the values of the class label *within* a given training set. If several objects *X* in a training set are all linked to a single object *y*, and if the class labels of the objects *X* are highly correlated, then it is more likely that a learning algorithm will create a model with components intended to predict precisely these instances than if only a single instance had this combination of attribute values and class label.

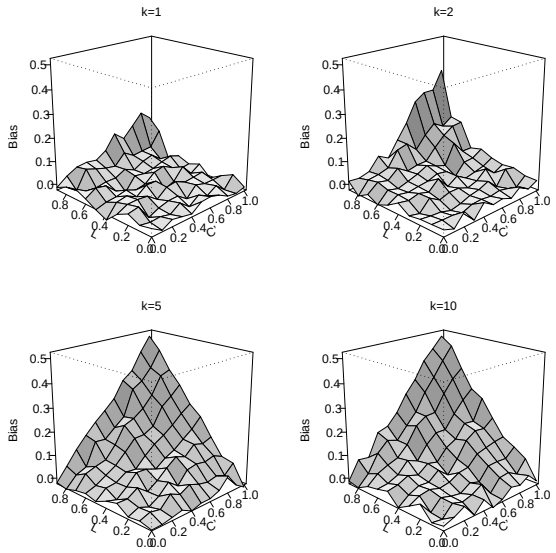

Fig. 5. Bias increases with linkage (*L*), autocorrelation (*C'*), and number of random attributes (*k*). Each point represents the average of 20 trials.

As noted above, relational autocorrelation represents an extremely important type of knowledge about relational data, one that can be exploited to improve accuracy (Neville and Jensen 2000; Slattery and Mitchell 2000). However, it can also fool algorithms and evaluation techniques not designed to account for its effects.

For example, consider the results shown in Figure 5. The graphs show how the bias varies across a wide range of linkage (*L*), autocorrelation (*C'*), and potential for overfitting. To alter this latter characteristic of learning algorithms, we varied the number of attributes *k* from one to ten. In each trial, we created synthetic data sets with 200 objects *X* and specified values of autocorrelation and single linkage with objects *Y*. Each object *x* was given a class label drawn from a binary uniform distribution. Each object *y* was given *k* attributes, each with a value drawn from a five-valued uniform distribution. This sample was then divided into equal-sized

training and test sets using simple random partitioning. We applied FOIL to the training set, and then evaluated the error of the resulting rules on the test set. For each combination of L, C', and k, we ran 20 trials and averaged the bias.

For a given number of attributes k, bias increases dramatically with increasing linkage L and autocorrelation C'. For high values of k, L, and C', bias is maximal (0.5). However, even moderate values of k, L, and C' produce substantial bias, confirming the results in the first section.

It is important to note that in the experiments presented above, these overfitted models are *not* learning any general knowledge about autocorrelation and linkage. For example, the FOIL rules learned for the experimental results depicted in Figure 4 contain only clauses of the form:

```
receipts(A) :- made-by(A,B), studio-attributes(B,C,D,E,F).
```

That is, they exclusively relate attributes of studios to *receipts* of movies linked to them. As noted previously, linkage and autocorrelation represent an important type of knowledge that could be exploited by a relational learning algorithm. However, most relational learning algorithms either cannot or do not learn probabilistic models of this form. The rules formed by FOIL on the synthetic data used in Figure 5 are of a similar form. The results in Figure 5 show that large bias that can result when algorithms learn overfitted models from data with strong linkage and autocorrelation.

Subgraph Sampling

Fortunately, this bias can be eliminated by a relatively small change to the procedure for creating training and test sets. *Subgraph sampling* guarantees that an object y and corresponding objects X appear only within a single subsample. This confines any autocorrelation among the class labels of objects X to a single subsample, and thus removes the dependence between subsamples due to concentrated linkage and relational autocorrelation.

We first introduced subgraph sampling of relational data in an earlier paper (Jensen and Neville 2001). However, we lacked a full understanding of the causes of dependence between subsamples, and we proposed an extreme form of subgraph sampling that eliminated all possible duplication of objects between subsamples, even when class labels were not autocorrelated through all types of objects. Here we propose a form of subgraph sampling that is far more conservative.

First, consider the special case where linkage and autocorrelation are high for only one type of object y, and that object exhibits single linkage with objects X. For example, in the movie data, only studios exhibit both high linkage and high autocorrelation; other types of objects (actors, directors, and producers) have fairly low values for one or both quantities. In addition, studios exhibit single linkage with movies. In this special case, we can partition a sample S based on the objects Y (e.g., studios), and then place all objects X in the same subsample as their corresponding y.

A more general partitioning algorithm first assigns objects X to prospective samples, and then incrementally converts prospective assignments to permanent assignments only if the corresponding objects Y for the given x are disjoint from other

objects Y already assigned to subsamples other than the prospective subsample of x. In contrast to the approach we proposed earlier (Jensen and Neville 2001), the objects Y considered during sampling should only be those through which linkage and autocorrelation is high.[3]

One feature of the general algorithm is worthy of special note — the random assignment of objects X to "prospective" subsamples. The algorithm either makes a prospective assignment permanent, or discards the object. An alternative algorithm would search for an assignment of objects to permanent subsamples that maximizes the number of objects assigned to each subsample, thus maximizing the size of subsamples. However, this approach can induce another form of statistical dependence among subsamples. Consider how such an "optimizing" algorithm would behave when confronted with a data set consisting of two disjoint (or nearly disjoint) sets of relational data. One subsample would be filled entirely with objects from one disjoint set, and another would be filled with objects from the other set. If the statistical characteristics of one of the disjoint sets did not mirror the characteristics of the other, then accuracy estimates of learned models would be biased downward.

Subgraph sampling resembles techniques that construct samples from a small number of completely disconnected graphs. For example, some experiments with WEBKB (Slattery and Mitchell 2000) 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) (Muggleton 1992). 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 can be created that preserve the statistical independence among samples.

Subgraph Sampling Eliminates Bias

In this section, we show how subgraph sampling eliminates the bias caused by linkage, autocorrelation, and overfitting. First, we replicate the experiments that produced Figure 4. However, rather than learning models for a randomized class label, we learn models on the original class label, but with samples produced by subgraph sampling. The results are shown in Figure 6. As before, the bias associated with simple random partitioning is high. However, the distribution of bias for subgraph sampling (b) has a mean bias near zero.

[3] What is considered "high" would vary somewhat by the desired precision of the estimated accuracy. This is a topic for future work.

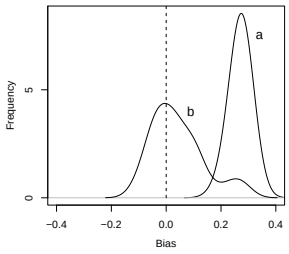

Fig. 6. Bias of FOIL models using random attributes for (a) simple random partitioning and (b) subgraph sampling.

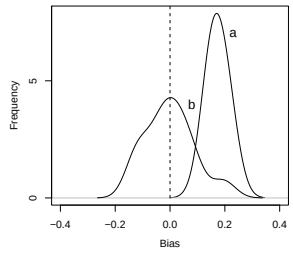

Fig. 7. Bias in FOIL models using non-random attributes for (a) simple random partitioning and (b) subgraph sampling.

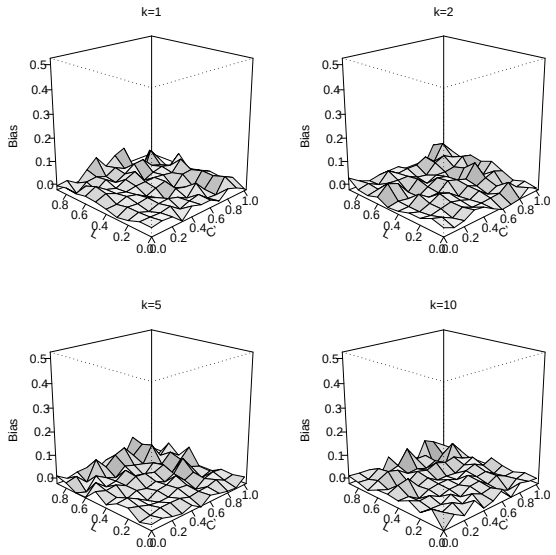

Fig. 8. Subgraph sampling with maximal separation eliminates bias at all levels of linkage (*L*), autocorrelation (*C'*), and number of random attributes (*k*).

The results in figures 4 and 6 were obtained using completely random attributes artificially generated on studios. However, similar results are obtained if we learn from attributes generated from the real characteristics of studios. The results in Figure 7 were generated by learning models with four attributes on studios. The attributes are the first letter of the studio name, the decade in which the studio was founded, the number of letters in the studio name (discretized to 10 unique values), and a binary attribute indicating whether the studio is located in the U.S. As before, the bias is high

for simple random partitioning. Bias for both distributions used the mean of the distribution for subgraph sampling as an estimate of true error. These results confirm that the bias does not result from some peculiarity in the generation of random attributes, but rather results from dependence between the training and test sets

These results only indicate bias for a single combination of L, C', and degree of overfitting. Figure 8 shows the results of more systematic variation of these quantities. The figure was produced from the same experiments as Figure 5, except the training and test sets were constructed by subgraph sampling rather than simple random partitioning. The result is extremely low bias across the full range of values of L, C', and k..

Conclusions and Future Work

Concentrated linkage and relational autocorrelation can cause strong bias in the test set accuracy of induced models. In this paper, we demonstrate the bias using FOIL, so that other researchers can easily replicate and extend our experiments, but we have also observed this phenomenon in our own algorithms for relational learning. Fortunately, the bias associated with linkage and autocorrelation can be corrected by using subgraph sampling in preference to simple random partitioning.

While some special classes of relational data naturally allow subgraph sampling, relational learning methods will increasingly encounter data in which this bias arises, as we extend our work to more general classes of relational data, including networks of web pages, bibliographic citations, financial transactions, messages, biochemical interactions, computers, supervisory relationships, and social interactions.

This work also emphasizes the need to pursue research on relational learning techniques that exploit relational autocorrelation to enhance the predictive power of relational models. Additional work should also investigate methods to estimate the bias associated with specific levels of autocorrelation and linkage, and to search for classes of objects that exhibit those degrees of linkage and autocorrelation, so more automated approaches to subgraph sampling can be devised.

Acknowledgments

Foster Provost and Hannah Blau provided valuable comments on an earlier version of this work and Ross Fairgrieve prepared the movie data for analysis. The data used in this paper were drawn from the Internet Movie Database (www.imdb.com) and several experiments used FOIL, by Ross Quinlan. This research is supported by The Defense Advanced Research Projects Agency (DARPA) and Air Force Research Laboratory (AFRL), Air Force Material Command, USAF, under agreements F30602-00-2-0597 and F30602-01-2-0566.

References

Cortes, C., D. Pregibon, and C. Volinsky (2001). Communities of Interest. *Proceedings Intelligent Data Analysis 2001.*

Craven, M., DiPasquo, D., Freitag, D., McCallum, A., Mitchell, T., Nigam, K., & Slattery, S. (1998). Learning to extract symbolic knowledge from the World Wide Web. *Proceedings of the Fifteenth National Conference on Artificial Intellligence (*pp. 509-516). Menlo Park: AAAI Press.

Cressie, N. (1993). *Statistics for Spatial Data*. Wiley.

Flach, P. and N. Lachiche (1999). 1BC: a first-order Bayesian classifier. *ILP'99*. 92-103. Springer.

Getoor, L., N. Friedman, D. Koller, and A. Pfeffer (1999). Learning probabilistic relational models. In *IJCAI'99*. 1300-1309.

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.

Jensen, D. and P. Cohen (2000). Multiple comparisons in induction algorithms. *Machine Learning* 38:309-338.

Jensen, D. and J. Neville (2001). Correlation and sampling in relational data mining. *Proceedings of the 33rd Symposium on the Interface of Computing Science and Statistics*.

Jensen, D. and J. Neville (2002). Linkage and autocorrelation cause bias in relational feature selection. *Machine Learning: Proceedings of the Nineteenth International Conference*. Morgan Kaufmann.

John, G., R. Kohavi, and K. Pfleger (1994). Irrelevant features and the subset selection problem. *ICML'94*. 121-129.

Kleinberg, J. (1999). Authoritative sources in a hyper-linked environment. *Journal of the ACM* 46:604-632,

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

Muggleton, S. (Ed) (1992). *Inductive Logic Programming*. Academic Press

Muggleton, S. (2000). Learning Stochastic Logic Programs. AAAI Workshop on Learning Statistical Models from Relational Data, 36-41.

Noreen, E. (1989). *Computer Intensive Methods for Testing Hypotheses*. Wiley.

Neville, J. and D. Jensen (2000). Iterative Classification in Relational Data. AAAI Workshop on Learning Statistical Models from Relational Data, 42-49.

Quinlan, J. R. (1990), Learning logical definitions from relations. *Machine Learning* 5:239-266.

Sachs, L. (1982). *Applied Statistics*. Springer-Verlag.

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.

Appendix

The movie data were coded for input to FOIL as follows: Each studio attribute was specified as an unordered discrete type with all attribute values flagged as a theory constants. Unordered type specifications also define movies and studios, with 1382 unique labels for the movies and 128 unique labels for the studios respectively.

The input contains one target relation and two background relations:

```
receipts(movie)
made-by(studio)
studio-attributes(studio, first-char, name-len, in-us, decade)
```

The target relation described above for movie receipts contains both positive and negative examples. The two background relations contain only positive examples; one

specifying the relationships between movies and studios, and the other specifying attribute values associated with each studio.

Learned clauses were similar in form to:

```
receipts(A) :- made-by(A,B), studio-attributes(B,C,D,E,F).
```

All experiments used the current version of FOIL (foil6.sh) obtained from: <http://www.cse.unsw.edu.au/~quinlan/>. Arguments to FOIL specified that negative literals were not to be considered and the minimum accuracy of any clause considered was at least 70%. Other than these two modifications, FOIL's default settings were used.