In Data Mining: Next Generation Challenges and Future Directions,
H. Kargupta, A. Joshi, K. Sivakumar, and Y. Yesha (Eds.), AAAI Press, 2004

Chapter 1

Relational Data Mining with Inductive Logic Programming for Link Discovery

Raymond J. Mooney[*], *Prem Melville*[*], *Lappoon Rupert Tang*[*], *Jude Shavlik*[‡], *Inês de Castro Dutra*[‡], *David Page*[‡], *Vítor Santos Costa*[‡]

[*]Department of Computer Sciences
University of Texas
Austin, TX 78712-1188
{mooney,melville,rupert}@cs.utexas.edu

[‡]Department of Biostatistics and Medical Informatics and
Department of Computer Sciences
University of Wisconsin
Madison, WI 53706-1685
{shavlik,dpage}@cs.wisc.edu, {dutra,vitor}@biostat.wisc.edu

Abstract:
Link discovery (LD) is an important task in data mining for counter-terrorism and is the focus of DARPA's Evidence Extraction and Link Discovery (EELD) research program. Link discovery concerns the identification of complex relational patterns that indicate potentially threatening activities in large amounts of relational data. Most data-mining methods assume data is in the form of a feature-vector (a single relational table) and cannot handle multi-relational data. *Inductive logic programming* is a form

of relational data mining that discovers rules in first-order logic from multi-relational data. This paper discusses the application of ILP to learning patterns for link discovery.

Keywords: Relational Data Mining, Inductive Logic Programming, counter-terrorism, link discovery

1.1 Introduction

Since the events of September 11, 2001, the development of information technology that could aid intelligence agencies in their efforts to detect and prevent terrorism has become an important focus of attention. The Evidence Extraction and Link Discovery (EELD) program of the Defense Advanced Research Projects Agency (DARPA) is one research project that attempts to address this issue. The establishment of the EELD program for developing advanced software for aiding the detection of terrorist activity pre-dates the events of 9/11. The program had its genesis at a preliminary DARPA planning meeting held at Carnegie Mellon University after the opening of the Center for Automated Learning and Discovery in June of 1998. This meeting discussed the possible formation of a new DARPA research program focused on novel knowledge-discovery and data-mining (KDD) methods appropriate for counter-terrorism.

The scope of the new program was subsequently expanded to focus on three related sub-tasks in detecting potential terrorist activity from numerous large information sources in multiple formats. *Evidence Extraction* (EE) is the task of obtaining structured evidence data from unstructured, natural-language documents. EE builds on information extraction technology developed under DARPA's earlier MUC (Message Understanding Conference) programs [Lehnert & Sundheim1991, Cowie & Lehnert1996] and the current ACE (Automated Content Extraction) program at the National Institute of Standards and Technology (NIST)[NIST2003]. *Link Discovery* (LD) is the task of identifying known, complex, multi-relational patterns that indicate potentially threatening activities in large amounts of relational data. It is therefore a form of pattern-matching that involves matching complex, multi-relational "patterns of interest" against large amounts of data. Some of the input data for LD comes from EE applied to news reports and other unstructured documents, other input data comes from existing relational databases on financial and other transactions. Finally, *Pattern Learning* (PL) concerns the automated discovery of new relational patterns for potentially threatening activities. Since determining and authoring a complete and accurate set of formal patterns for detecting terrorist activities is a difficult task, learning methods may be useful for automatically acquiring such patterns from supervised or unsupervised data. Learned patterns can then be employed by an LD system to improve its detection of threatening activities. The current EELD program focused on these three sub-topics started in the summer of 2001. After 9/11, it was incorporated under the new Information Awareness Office (IAO) at DARPA.

The data and patterns used in EELD include representations of people, organizations, objects, and actions and many types of relations between them. The data is perhaps best represented as a large graph of entities connected by a variety of relations.

The areas of *link analysis* and *social network analysis* in sociology, criminology, and intelligence [Jensen & Goldberg1998, Wasserman & Faust1994, Sparrow1991] study such networks using graph-theoretic representations. Data mining and pattern learning for counter terrorism therefore requires handling such multi-relational, graph-theoretic data.

Unfortunately, most current data-mining methods assume the data is from a single relational table and consists of flat tuples of items, as in market-basket analysis. This type of data is easily handled by machine learning techniques that assume a "propositional" (a.k.a "feature vector" or "attribute value") representation of examples [Witten & Frank1999]. *Relational data mining* (RDM) [Džeroski & Lavrač2001b], on the other hand, concerns mining data from multiple relational tables that are richly connected. Given the style of data needed for link discovery, pattern learning for link discovery requires *relational* data mining. The most widely studied methods for inducing relational patterns are those in *inductive logic programming* (ILP) [Muggleton1992, Lavrac & Dzeroski1994]. ILP concerns the induction of Horn-clause rules in first-order logic (i.e., logic programs) from data in first-order logic. This paper discusses our on-going work on applying ILP to pattern learning for link discovery as a part of the EELD project.

1.2 Inductive Logic Programming (ILP)

ILP is the study of learning methods for data and rules that are represented in first-order predicate logic. Predicate logic allows for quantified variables and relations and can represent concepts that are not expressible using examples described as feature vectors. A relational database can be easily translated into first-order logic and be used as a source of data for ILP [Wrobel2001]. As an example, consider the following rules, written in Prolog syntax (where the conclusion appears first), that define the uncle relation:

```
uncle(X,Y) :- brother(X,Z),parent(Z,Y).
uncle(X,Y) :- husband(X,Z),sister(Z,W),parent(W,Y).
```

The goal of *inductive logic programming* (ILP) is to infer rules of this sort given a database of background facts and logical definitions of other relations [Muggleton1992, Lavrac & Dzeroski1994]. For example, an ILP system can learn the above rules for uncle (the *target predicate*) given a set of positive and negative examples of uncle relationships and a set of facts for the relations parent, brother, sister, and husband (the *background predicates*) for the members of a given extended family, such as:

```
uncle(tom,frank), uncle(bob,john),
not uncle(tom,cindy), not uncle(bob,tom)
parent(bob,frank), parent(cindy,frank), parent(alice,john),
parent(tom,john), brother(tom,cindy), sister(cindy,tom),
husband(tom,alice), husband(bob,cindy).
```

Alternatively, rules that logically define the brother and sister relations could be supplied and these relationships inferred from a more complete set of facts about only the "basic" predicates: `parent`, `spouse`, and `gender`.

If-then rules in first-order logic are formally referred to as *Horn clauses*. A more formal definition of the ILP problem follows:

- **Given:**
  - Background knowledge, B, a set of Horn clauses.
  - Positive examples, P, a set of Horn clauses (typically ground literals).
  - Negative examples, N, a set of Horn clauses (typically ground literals).
- **Find:** A hypothesis, H, a set of Horn clauses such that:
  - $\forall p \in P : H \cup B \models p$ (completeness)
  - $\forall n \in N : H \cup B \not\models n$ (consistency)

A variety of algorithms for the ILP problem have been developed [Džeroski & Lavrač2001a] and applied to a variety of important data-mining problems [Džeroski2001, Houstis *et al.*2000]. Nevertheless, relational data mining remains an under-appreciated topic in the larger KDD community. For example, recent textbooks on data mining [Han & Kamber2001, Witten & Frank1999, Hand, Mannila, & Smyth2001] hardly mention the topic. An increasing number of applications require handling complex, structured data types, including bioinformatics, web and text mining, and engineering. Therefore, we believe it is an important topic for "next generation" data mining systems. In particular, it is critical for link discovery applications in counter-terrorism.

One of the standard criticisms of ILP methods from a data-mining perspective is that they do not scale to large amounts of data. Since the hypothesis space of possible logic programs is extremely large and since just testing individual hypotheses requires potentially complex automated deduction, ILP methods can have difficulty processing large amounts of data. We have developed methods to help address both of these aspects of computational complexity. First, we have developed methods for controlling the number of hypotheses tested by developing new search methods that use stochastic search to more efficiently explore the space of hypotheses [Zelezny, Srinivasan, & Page2002] or that combine aspects of existing top-down and bottom-up methods (see section 1.3.2). Second, we have developed methods for automatically optimizing learned clauses by inserting cuts[1] in the Prolog code so that deduction is more efficient [Santos Costa, Srinivasan, & Camacho2000]. However, as discussed in section 1.4, scaling ILP to very large data sets is a significant area for future research.

1.3 Initial Work on ILP for Link Discovery

We tested several ILP algorithms on various EELD datasets. The current EELD datasets pertain to two domains that were chosen as "challenge problems" in link discovery that

[1]In Prolog, cuts (!) are procedural operators that prevent potentially computationally expensive backtracking where the programmer determines it is unnecessary.

have many of the underlying properties of the counter-terrorism problem – Nuclear Smuggling and Contract Killing. The Contract-Killing domain is further divided into natural (real world) data manually collected and extracted from news sources and synthetic (artificial) data generated by simulators. Section 1.3.1 presents our experimental results on the natural Smuggling and Contract-Killing data, while section 1.3.2 presents results on the synthetic Contract-Killing data.

1.3.1 Experiments on Natural Data

The Nuclear-Smuggling Data

The Nuclear-Smuggling dataset consists of reports on Russian nuclear materials smuggling [McKay, Woessner, & Roule2001]. The Chronology of Nuclear and Radioactive Smuggling Incidents is the basis for the analysis of patterns in the smuggling of Russian nuclear materials. The information in the Chronology is based on open-source reporting, primarily World News Connection (WNC) and Lexis-Nexis. There are also some articles obtained from various sources that have been translated from Italian, German and Russian. The research from which the Chronology grew began in 1994 and the chronology itself first appeared as an appendix to a paper by Williams and Woessner in 1995 [Williams & Woessner1995b, Williams & Woessner1995a]. The continually evolving Chronology then was published twice as separate papers in the same journal as part of the "Recent Events" section [Woessner1995, Woessner1997]. As part of the EELD project, the coverage of the Chronology was extended to March 2000 and grew to include 572 incidents.

The data is provided in the form of a relational database. This database contains Objects (described in rows in tables), each of which has Attributes of differing types (i.e., columns in the tables), the values of which are provided by the source information or the analyst. The Objects are of different types, which are denoted by prefixes (E_, EV_, and LK_), and consist of the following:

- Entity Objects (E_...): these consist of E_LOCATION, E_MATERIAL, E_ORGANIZATION, E_PERSON, E_SOURCE, and E_WEAPON;
- Event Objects (EV_...): these currently consist of the generic EV_EVENT;
- Link Objects (LK_...): used for expressing links between/among Entities and Events,

The database has over 40 relational tables. The number of tuples in a relational table varies from as many as 800 to as little as 2 or 3.

As a representative problem, we used ILP to learn rules for determining which events in an incident are *linked*. Such rules could be used to construct larger knowledge structures that could be recognized as threats. Hence, the ILP system is given positive training examples of known "links" between events. We assume all other events are unrelated and therefore compose a set of negative examples. We also provide background knowledge that the *linked* relation is commutative. Our training set consists of 140 positive examples and 140 distinct negative examples randomly drawn from a full set of 8,124 negative pairs of events. The linking problem in the Nuclear-Smuggling data is thus quite challenging in that it is a heavily relational learning problem over a large number of relations, whereas traditional ILP applications usually require a small number of relations.

The Natural Contract-Killing Data

The dataset of contract killings was first compiled by O'Hayon and Cook [Cook & O'Hayon2000] in response to research on Russian organized crime that encountered frequent references to contract killings. The dataset was subsequently expanded by the authors with funding from the EELD program through Veridian Systems Division [Williams2002]. The database consists of a chronology of incidents each described using information drawn from one or more news articles. As in the Nuclear-Smuggling dataset, information in the chronology is based on open-source reporting, especially Foreign Broadcast Information Service (FBIS) and Joint Publications Research Service (JPRS) journals, and subsequently both FBIS on-line and the on-line version World News Connection (WNC). These services and Lexis-Nexis were the main information sources.

The data is organized in relational tables in the same format as the Nuclear-Smuggling data described in the previous section. The dataset used in our experiments has 48 relational tables. The number of tuples in a relational table varies from as many as 1,000 to as few as 1. Each killing was categorized according to one of three possible motivations: "Rival," "Obstacle," or "Threat." The ILP task was to determine whether the motivation for a killing was categorized as "Rival" or not. The motivation for this learning task was to recognize patterns of activity that indicate an underlying motive, which in turn contributes to recognizing threats. The number of positive examples in this dataset is 38, while the number of negative examples is 34.

ILP Results on the Natural Data

ALEPH We used the ILP system ALEPH [Srinivasan2001] to learn rules for the natural datasets. By default, ALEPH uses a simple greedy set covering procedure that constructs a complete and consistent hypothesis one clause at a time. In the search for any single clause, ALEPH selects the first uncovered positive example as the seed example, *saturates* this example, and performs an admissible search over the space of clauses that subsume this saturation, subject to a user-specified clause length bound. Further details about our use of ALEPH in these experiments are available from [de Castro Dutra *et al.*2002].

Ensembles *Ensembles* aim at improving accuracy through combining the predictions of multiple classifiers in order to obtain a single classifier. In contrast with previous approaches [Quinlan1996, Hoche & Wrobel2001], each classifier is a logical theory generated by ALEPH. Many methods have been presented for ensemble generation [Dietterich1998]. We use *bagging* [Breiman1996a], a popular method that is known to frequently create a more accurate ensemble. Bagging works by training each classifier on a random sample from the training set. Bagging has the important advantage that it is effective on "unstable learning algorithms" [Breiman1996b], where small variations in the input data can cause large variations in the learned theories. Most ILP algorithms are unstable in this sense. A second advantage is that the bagging algorithm is highly

```
linked(A,E) :-
   lk_event_person(_,EventA,PersonC,_,RelationB,RelationB,DescriptionD),
   lk_event_person(_,EventF,PersonC,_,RelationB,RelationB,DescriptionD),
   lk_material_location(_,MaterialG,_,EventE,_,_,_,_,_),
   lk_event_material(_,EventF,MaterialG,_,_,_,_).
```

Figure 1.1: Nuclear-Smuggling Data: Sample Learned Rule

parallel [Dutra *et al.*2003]. Further details about our approach to bagging for ILP, as well as our experimental methodology, can be found in [de Castro Dutra *et al.*2002]. Our experimental results are based on a five-fold cross-validation, where five times we train on 80% of the examples and then test what was learned on the remaining 20% (in addition, each example is in one and only one test set).

For the task of identifying linked events in the Nuclear-Smuggling dataset, ALEPH produces an average testset accuracy of 85%. This is an improvement over the baseline case (majority class—always guessing two events are not linked), which produces an average accuracy of 78%. Bagging (with 25 different sets of rules) increases the accuracy to almost 90%.

An example of a rule with good accuracy found by the system is shown in Figure 1.1. This rule covers 43 of the 140 positive examples and no negative examples. According to this rule, two smuggling events `A` and `E` are related if event `A` involves a person `C` who is also involved in another event `F`. Event `F` involves some material `G` that appears in event `E`. In other words, a person `C` in event `A` is involved in a third event `F` that uses material from event `E`. Person `C` played the same role `B`, with description `D`, in events `A` and `F`. The "_" symbols mean that those arguments were not relevant for that rule. Figure 1.2 illustrates the connections between events, material and people involved. Solid lines are direct connections shown by the literals in the body of the clause. The dotted line corresponds to the newly learned concept that describes a connection between two events.

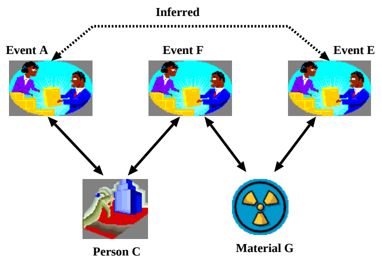

Figure 1.2: Pictorial representation of a learned rule.

The task of identifying the underlying motive in the Contract-Killing data set is much more difficult, with ALEPH's accuracy at 59%, compared with the baseline accuracy of 52%. Again, bagging improves the accuracy, this time to 69%. The rule in Figure 1.3 shows one logical clause the ILP system found for this dataset. The rule covers 19 of the 38 positive examples and a single negative example. The rule says that

```
rivalKilling(EventA) :-
     lk_event_event(_,EventB,EventA,RelationC,EventDescriptionD),
     lk_event_event(_,EventB,EventE,RelationC,EventDescriptionD),
     lk_event_event(_,EventE,EventF,_,EventDescriptionD),
     lk_org_org(_,_,_,EventF,_,_,_,_,_).
```

Figure 1.3: Natural Contract-Killing Data: Sample Learned Rule

event `A` is a killing by a rival if we can follow a chain of events that connects event `A` to event `B`, event `B` to event `E`, and event `E` to an event `F` that relates two organizations. Events `A` and `E` have the same kind of relation, `RelationC`, to `B`. All events in the chain are subsets of the same incident `D`.

1.3.2 Experiments on Synthetic Data

The ease of generating large amounts and data and privacy considerations have led the EELD program to use synthetic data generated by simulators. Next, we describe the results we obtained from simulated data for the CK problem. This data was generated from a run of a Task-Based (TB) simulator developed by Information Extraction and Transport Incorporated (IET). The TB simulator outputs case files, which contain complete and unadulterated descriptions of murder cases. These case files are then filtered for observability, so that facts that would not be accessible to an investigator are eliminated. To make the task more realistic, the simulator output is corrupted, e.g., by misidentifying role players or incorrectly reporting group memberships. This filtered and corrupted data form the evidence files. In the evidence files, facts about each event are represented as ground facts, such as:

```
murder(Murder714)
perpetrator(Murder714, Killer186)
crimeVictim(Murder714, MurderVictim996)
deviceTypeUsed(Murder714, PistolCzech)
```

The synthetic dataset that we used consists of 632 murder events. Each murder event has been labeled as either a positive or negative example of a murder-for-hire. There are 133 positive and 499 negative examples in the dataset. Our task was to learn a theory to correctly classify an unlabeled event as either a positive or negative instance of murder-for-hire. The amount of background knowledge for this dataset is extremely large; consisting of 52 distinct predicate names, and 681,039 background facts in all.

Scaling to large datasets in data mining typically refers to increasing the *number* of training examples that can be processed. Another measure of complexity that is particularly relevant in relational data mining is the *size* of individual examples, i.e. the number of facts used to describe each example. To our knowledge, the challenge problems developed for the EELD program are the largest ILP problems attempted to date in terms of the number of facts in the background knowledge. In order to more effectively process such large examples, we have developed a new ILP method that reduces the number of clauses that are generated and tested.

BETH

The two standard approaches to ILP are bottom-up and top-down [Lavrac & Dzeroski1994]. Bottom-up methods like ALEPH start with a very specific clause (called a *bottom clause*) generated from a seed positive example and generalize it as far as possible without covering negative examples. Top-down methods like FOIL [Quinlan1990] and mFOIL [Lavrac & Dzeroski1994] start with the most general (empty) clause and repeatedly specialize it until it no longer covers negative examples. Both approaches have problems scaling to large examples. When given large amounts of background knowledge, the bottom clause in bottom-up methods becomes intractably large and the increased branching factor in top-down methods greatly impedes their search.

Since top-down and bottom-up approaches have both strengths and weaknesses, we developed a hybrid method that helps reduce search when learning with large amounts of background knowledge. It does not build a bottom clause using a seed example *before* searching for a good clause. Instead, after a random seed example is chosen, it generates literals in a top-down fashion (i.e. guided by heuristic search), except the literals generated are constrained to cover the seed example. Based on this idea, we have developed a system called **B**ottom-clause **E**xploration **T**hrough **H**euristic-search (BETH) in which the bottom clause is not constructed in advance but "discovered" during the search for a good clause. Details of the algorithm are given in [Tang, Mooney, & Melville2003].

Results and Discussion

The performance of ALEPH, mFOIL, and BETH was evaluated using 6-fold cross-validation. The data for each fold was generated by separate runs of the TB simulator. The facts produced by one run of the simulator, only pertain to the entities and relations generated in that run; hence the facts of each fold are unrelated to the others. For each trial, one fold is set aside for testing, while the remaining data is *combined* for training. The total number of Prolog atoms in the data is so large that running more than six folds is not feasible.[2] To test performance on varying amounts of training data, learning curves were generated by testing the system after training on increasing subsets of the overall training data. Note that, for different points on the learning curve, the background knowledge remains the same; only the number of positive and negative training examples given to the system varies.

We compared the three systems with respect to accuracy and training time. Accuracy is defined as the number of correctly classified test cases divided by the total number of test cases. The training time is measured as the CPU time consumed during the training phase. All the experiments were performed on a 1.1 GHz Pentinum with dual processors and 2 GB of RAM. BETH and mFOIL were implemented in Sicstus Prolog version 3.8.5 and ALEPH was implemented in Yap version 4.3.22. Although different Prolog compilers were used, the Yap Prolog compiler has been demonstrated to outperform the Sicstus Prolog compiler, particularly in ILP applications [Santos Costa1999].

The following is a sample rule learned by BETH:

[2]The maximum number of atoms that the Sicstus Prolog compiler can handle is approximately a quarter million.

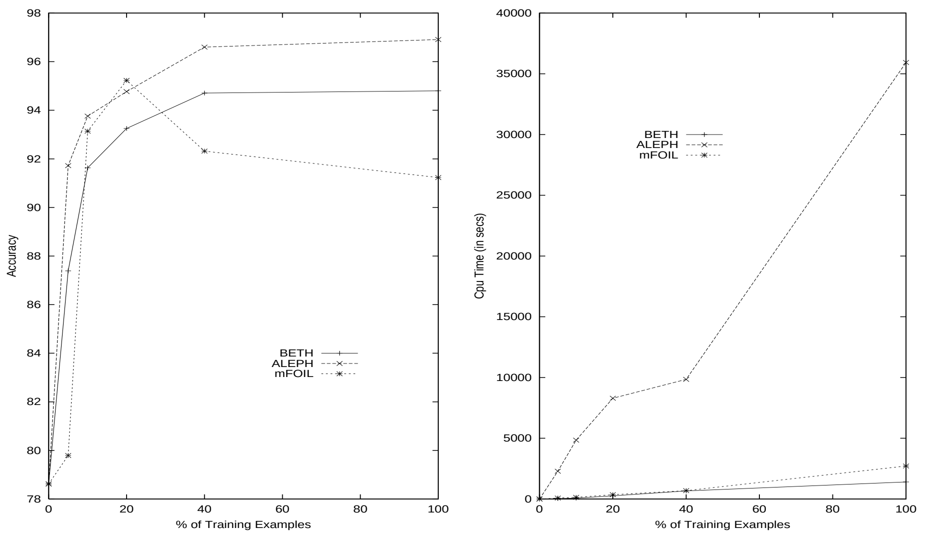

Figure 1.4: Learning Curves for Accuracy and Training Speed

System	*Accuracy*	*CPU Time (mins)*	*# of Clauses*	*Bottom Clause Size*
BETH	94.80% (+/- 2.3%)	23.39 (+/- 4.26)	4483	34
ALEPH	96.91% (+/- 2.8%)	598.92 (+/- 250.00)	63334	4061
mFOIL	91.23% (+/- 4.8%)	45.28 (+/- 5.40)	112904	n/a

Table 1.1: Results on classifying *murder-for-hire* events given all the training data. *# of Clauses* is the total number of clauses tested; and *Bottom Clause Size* is the average number of literals in the bottom clause constructed for each clause in the learned theory. The 90% confidence intervals are given for test *Accuracy* and *CPU time*.

```
murder_for_hire(A):- murder(A), eventOccursAt(A,H),
  geographicalSubRegions(I,H), perpetrator(A,B),
  recipientOfinfo(C,B), senderOfinfo(C,D), socialParticipants(F,D),
  socialParticipants(F,G), payer(E,G), toPossessor(E,D).
```

This rule covered 9 positive examples and 3 negative examples. The rule can be interpreted as: *A* is a murder-for-hire, if *A* is a murder event, which occurs in a city in a subregion of Russia, and in which *B* is the perpetrator, who received information from *D*, who had a meeting with and received some money from *G*.

The results of our experiments are summarized in Figure 1.4. A snapshot of the performance of the three ILP systems given 100% of the training examples is shown in Table 1.1. On the full training set, BETH trains 25 times faster than ALEPH while losing only 2 percentage points in accuracy and it trains twice as fast as mFOIL while gaining 3 percentage points in accuracy. Therefore, we believe that its integration of top-down and bottom-up search is an effective approach to dealing with the problem of scaling ILP to large examples. The learning curves for training time further illustrate that although BETH and mFOIL appear to scale linearly with the number of training examples, ALEPH's training-time growth is super-linear.

Systems like BETH and ALEPH construct literals based on actual ground atoms in the background knowledge, guaranteeing that the specialized clause covers at least the seed example. On the other hand, mFOIL generates more literals than necessary by enumerating all possible combination of variables. Some such combinations make useless literals; adding any of them to the body of the current clause makes specialized clauses that do not cover any positive examples. Thus, mFOIL wastes CPU time constructing and testing these literals. Since the average predicate arity in the EELD data was small (2), the speedup over mFOIL was not as great, although much larger gains would be expected for data that contains predicates with higher arity.

1.4 Current and Future Research

An under-studied issue in relational data mining is scaling algorithms to very large databases. Most research on ILP and RDM has been conducted in the machine learning and artificial intelligence communities rather than in the database and systems communities. Consequently, there has been insufficient research on systems issues involved in performing RDM in commercial relational-database systems and scaling algorithms to extremely large datasets that will not fit in main memory. Integrating ideas from systems work in data mining and deductive databases [Ramamohanarao & Harland1994] would seem to be critical in addressing these issues.

On the issue of scaling, in addition to the BETH system discussed in section 1.3.2, we are currently working on efficiently learning complex relational concepts from large amounts of data by using stochastic sampling methods. A major shortcoming of ILP is the computational demand that results from the large hypothesis spaces searched. Intelligently sampling these large spaces can provide excellent performance in much less time [Srinivasan1999, Zelezny, Srinivasan, & Page2002].

We are also developing algorithms that learn more robust, probabilistic relational concepts represented as stochastic logic programs [Muggleton2003] and variants. This will enrich the expressiveness and robustness of learned concepts. As an alternative to stochastic logic programs, we are working on learning clauses in a constraint logic programming language where the constraints are Bayesian networks [Page2000, Costa *et al.*2003].

One approach that we plan to investigate further is the use of approximate prior knowledge to induce more accurate, comprehensible relational concepts from fewer training examples [Richards & Mooney1995]. The use of prior knowledge can greatly reduce the burden on users; they can express the "easy" aspects of the task at hand and then collect a small number of training examples to refine and extend this prior knowledge.

We also plan to use active learning to allow our ILP systems to select more effective training examples for interactively learning relational concepts [Muggleton *et al.*1999]. By intelligently choosing the examples for users to label, better extraction accuracy can be obtained from fewer examples, thereby greatly reducing the burden on the users of our ILP systems.

Another important issue related to data mining for counter-terrorism is privacy preservation. DARPA's counter-terrorism programs have attracted significant public and media attention due to concerns about potential privacy violations (e.g. [Clymer2003]). Consequently, privacy-preserving data mining [Gehrke2002] is another very significant "next generation" issue in data mining.

1.5 Related Work

Although it is the most widely studied, ILP is not the only approach to relational data mining. In particular, other participants in the EELD program are taking alternative RDM approaches to pattern learning for link discovery. This section briefly reviews these other approaches.

1.5.1 Graph-based Relational Learning

Some relational data mining methods are based on learning structural patterns in graphs. In particular, SUBDUE [Cook & Holder1994, Cook & Holder2000] discovers highly repetitive subgraphs in a labeled graph using the minimum description length (MDL) principle. SUBDUE can be used to discover interesting substructures in graphical data as well as to classify and cluster graphs. Discovered patterns do not have to match the data exactly since SUBDUE can employ an inexact graph-matching procedure based on graph edit-distance. SUBDUE has been successfully applied to a number of important RDM problems in molecular biology, geology, and program analysis. It is also currently being applied to discover patterns for link discovery as a part of the EELD project (more details at `http://ailab.uta.edu/eeld/`). Since relational data for LD is easily represented as labeled graphs, graph-based RDM methods like SUBDUE are a natural approach.

1.5.2 Probabilistic Relational Models

Probabilistic relational models (PRM's) [Koller & Pfeffer1998] are an extension of Bayesian networks for handling relational data. Methods for learning Bayesian networks have also been extended to produce algorithms for inducing PRM's from data [Friedman *et al.*1999]. PRM's have the nice property of integrating some of the advantages of both logical and probabilistic approaches to knowledge representation and reasoning. They combine some of the representational expressivity of first-order logic with the uncertain reasoning abilities of Bayesian networks. PRM's have been applied to a number of interesting problems in molecular biology, web-page classification, and analysis of movie data. They are also currently being applied to pattern learning for link discovery as a part of the EELD project.

1.5.3 Relational Feature Construction

One approach to learning from relational data is to first "flatten" or "propositionalize" the data by constructing features that capture some of the relational information and then applying a standard learning algorithm to the resulting feature vectors [Kramer, Lavrač, & Flach2001]. PROXIMITY [Neville & Jensen2000] is a system that constructs features for categorizing entities based on the categories and other properties of other entities to which it is related. It then uses an interactive classification procedure to dynamically update inferences about objects based on earlier inferences about related objects. PROXIMITY has been successfully applied to company and movie data. It is also currently being applied to pattern learning for link discovery as a part of the EELD project.

1.6 Conclusions

Link discovery is an important problem in automatically detecting potential threatening activity from large, heterogeneous data sources. The DARPA EELD program is a U.S. government research project exploring link discovery as an important problem in the development of new counter-terrorism technology. Learning new link-discovery patterns that indicate potentially threatening activity is a difficult data mining problem. It requires discovering novel relational patterns in large amounts of complex relational data. In this work we have shown that ILP methods can extract interesting and useful rules from link-discovery data-bases containing up to hundreds of thousands of items. To do so, we improved search efficiency and computation time per node over current ILP systems.

Most existing data-mining methods assume flat data from a single relational table and are not appropriate for link discovery. Relational data mining techniques, such as inductive logic programming, are needed. Many other problems in molecular biology [Srinivasan *et al.*1996], natural-language understanding [Zelle & Mooney1996], web page classification [Craven *et al.*2000], information extraction [Califf & Mooney1999, Freitag1998], and other areas also require mining multi-relational data. However, relational data mining requires exploring a much larger space of possible patterns and performing complex inference and pattern matching. As a result, current RDM methods are not sufficiently scalable to very large databases. Consequently, we believe that relational data mining is one of the major research topics in the development of the next generation of data mining systems, particularly those in the area of counter-terrorism.

Acknowledgments

This research is sponsored by the Defense Advanced Research Projects Agency and managed by Rome Laboratory under contract F30602-01-2-0571. The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied of the Defense Advanced Research Projects Agency, Rome Laboratory, or the United States Government.

Vítor Santos Costa and Inês de Castro Dutra are on leave from COPPE/Sistemas, Federal University of Rio de Janeiro and were partially supported by CNPq. Many thanks to Hans Chalupksy's group at ISI, in particular to André Valente, who gave us support on using the Task-based simulator. We would like to thank the Biomedical Computing Group support staff and the Condor Team at the Computer Sciences Department of the University of Wisconsin, Madison, for their invaluable help with Condor. We also would like to thank Ashwin Srinivasan for his help with the ALEPH system.

Bibliography

[Breiman1996a] Breiman, L. 1996a. Bagging Predictors. *Machine Learning* 24(2):123–140.

[Breiman1996b] Breiman, L. 1996b. Stacked Regressions. *Machine Learning* 24(1):49–64.

[Califf & Mooney1999] Califf, M. E., and Mooney, R. J. 1999. Relational learning of pattern-match rules for information extraction. In *Proceedings of the 17th National Conference on Artificial Intelligence*, 328–334.

[Clymer2003] Clymer, A. 2003. Pentagon Surveillance Plan Is Described as Less Invasive. *New York Times* May(7).

[Cook & Holder1994] Cook, D. J., and Holder, L. B. 1994. Substructure discovery using minimum description length and background knowledge. *Journal of Artificial Intelligence Research* 1:231–255.

[Cook & Holder2000] Cook, D. J., and Holder, L. B. 2000. Graph-based data mining. *IEEE Intelligent Systems* 15(2):32–41.

[Cook & O'Hayon2000] Cook, W., and O'Hayon, G. 2000. Chronology of Russian killings. *Transnational Organized Crime* 4(2).

[Costa *et al.*2003] Costa, V. S.; Page, D.; Qazi, M.; and Cussens, J. 2003. CLP(BN): Constraint logic programming for probabilistic knowledge. In *Proceedings of the International Conference on Uncertainty in Artificial Intelligence (UAI-03)*.

[Cowie & Lehnert1996] Cowie, J., and Lehnert, W. 1996. Information extraction. *Communications of the ACM* 39(1):80–91.

[Craven *et al.*2000] Craven, M.; DiPasquo, D.; Freitag, D.; McCallum, A. K.; Mitchell, T.; Nigam, K.; and Slattery, S. 2000. Learning to construct knowledge bases from the World Wide Web. *Artificial Intelligence* 118(1-2):69–113.

[de Castro Dutra *et al.*2002] de Castro Dutra, I.; Page, D.; Costa, V. S.; and Shavlik, J. W. 2002. An empirical evaluation of bagging in inductive logic programming. In *Inductive Logic Programming, 12th International Conference*, volume 2583 of *Lecture Notes in Computer Science*, 48–65. Sydney, Australia: Springer Verlag.

[Dietterich1998] Dietterich, T. G. 1998. Machine-learning research: Four current directions. *The AI Magazine* 18(4):97–136.

[Dutra *et al.*2003] Dutra, I. C.; Page, D.; Santos Costa, V.; Shavlik, J. W.; and Waddell, M. 2003. Towards automatic management of embarassingly parallel applications. In *Proceedings of Europar 2003*, Lecture Notes in Computer Science. Klagenfurt, Austria: Springer Verlag.

[Džeroski & Lavrač2001a] Džeroski, S., and Lavrač, N. 2001a. An introduction to inductive logic programming. In Džeroski, S., and Lavrač, N., eds., *Relational Data Mining*. Berlin: Springer Verlag. 48–73.

[Džeroski & Lavrač2001b] Džeroski, S., and Lavrač, N., eds. 2001b. *Relational Data Mining*. Berlin: Springer Verlag.

[Džeroski2001] Džeroski, S. 2001. Relational data mining applications: An overview. In Džeroski, S., and Lavrač, N., eds., *Relational Data Mining*. Berlin: Springer Verlag. 339–364.

[Freitag1998] Freitag, D. 1998. Information extraction from HTML: Application of a general learning approach. In *Proceedings of the 16th National Conference on Artificial Intelligence*, 517–523. Madison, WI: AAAI Press / The MIT Press.

[Friedman *et al.*1999] Friedman, N.; Getoor, L.; Koller, D.; and Pfeffer, A. 1999. Learning probabilistic relational models. In *Proceedings of the 16th International Joint Conference on Artificial Intelligence*, 1300–1307.

[Gehrke2002] Gehrke, J. 2002. Data mining for security and privacy. *SIGKDD Explorations* 4(2):i. Introduction to special issue on Privacy and Security.

[Han & Kamber2001] Han, J., and Kamber, M. 2001. *Data Mining: Concepts and Techniques*. San Francisco: Morgan Kauffmann Publishers.

[Hand, Mannila, & Smyth2001] Hand, D. J.; Mannila, H.; and Smyth, P. 2001. *Principles of Data Mining*. Cambridge, MA: MIT Press.

[Hoche & Wrobel2001] Hoche, S., and Wrobel, S. 2001. Relational learning using constrained confidence-rated boosting. In Rouveirol, C., and Sebag, M., eds., *Proceedings of the 11th International Conference on Inductive Logic Programming*, volume 2157 of *Lecture Notes in Artificial Intelligence*, 51–64. Springer-Verlag.

[Houstis *et al.*2000] Houstis, E. N.; Catlin, A. C.; Rice, J. R.; Verykios, V. S.; Ramakrishnan, N.; and Houstis, C. E. 2000. PYTHIA-II: a knowledge/database system for managing performance data and recommending scientific software. *ACM Transactions on Mathematical Software* 26(2):227–253.

[Jensen & Goldberg1998] Jensen, D., and Goldberg, H., eds. 1998. *AAAI Fall Symposium on Artificial Intelligence for Link Analysis*. Menlo Park, CA: AAAI Press.

[Koller & Pfeffer1998] Koller, D., and Pfeffer, A. 1998. Probabilistic frame-based systems. In *Proceedings of the 16th National Conference on Artificial Intelligence*, 580–587. Madison, WI: AAAI Press / The MIT Press.

[Kramer, Lavrač, & Flach2001] Kramer, S.; Lavrač, N.; and Flach, P. 2001. Propositionalization approaches to relational data mining. In Džeroski, S., and Lavrač, N., eds., *Relational Data Mining*. Berlin: Springer Verlag. 262–291.

[Lavrac & Dzeroski1994] Lavrac, N., and Dzeroski, S. 1994. *Inductive Logic Programming: Techniques and Applications*. Ellis Horwood.

[Lehnert & Sundheim1991] Lehnert, W., and Sundheim, B. 1991. A performance evaluation of text-analysis technologies. *AI Magazine* 12(3):81–94.

[McKay, Woessner, & Roule2001] McKay, S. J.; Woessner, P. N.; and Roule, T. J. 2001. Evidence extraction and link discovery (EELD) seedling project, database schema description, version 1.0. Technical Report 2862, Veridian Systems Division.

[Muggleton *et al.*1999] Muggleton, S.; Bryant, C.; Page, C.; and Sternberg, M. 1999. Combining active learning with inductive logic programming to close the loop in machine learning. In Colton, S., ed., *Proceedings of the AISB'99 Symposium on AI and Scientific Creativity (informal proceedings)*.

[Muggleton1992] Muggleton, S. H., ed. 1992. *Inductive Logic Programming*. New York, NY: Academic Press.

[Muggleton2003] Muggleton, S. 2003. Stochastic logic programs. *Journal of Logic Programming*. To appear.

[Neville & Jensen2000] Neville, J., and Jensen, D. 2000. Iterative classification in relational data. In *Papers from the AAAI-00 Workshop on Learning Statistical Models from Relational Data*. Austin, TX: AAAI Press / The MIT Press.

[NIST2003] NIST. 2003. ACE - Automatic Content Extraction. http://www.nist.gov/speech/tests/ace/.

[Page2000] Page, D. 2000. ILP: Just do it! In Lloyd, J.; Dahl, V.; Furbach, U.; Kerber, M.; Lau, K.-K.; Palamidessi, C.; Pereira, L.; Sagiv, Y.; and Stuckey, P., eds., *Proceedings of Computational Logic 2000*, 25–40. Springer Verlag.

[Quinlan1990] Quinlan, J. R. 1990. Learning logical definitions from relations. *Machine Learning* 5(3):239–266.

[Quinlan1996] Quinlan, J. R. 1996. Boosting first-order learning. *Algorithmic Learning Theory, 7th International Workshop, Lecture Notes in Computer Science* 1160:143–155.

[Ramamohanarao & Harland1994] Ramamohanarao, K., and Harland, J. 1994. An introduction to deductive database languages and systems. *VLDB Journal* 3:2.

[Richards & Mooney1995] Richards, B. L., and Mooney, R. J. 1995. Automated refinement of first-order Horn-clause domain theories. *Machine Learning* 19(2):95–131.

[Santos Costa, Srinivasan, & Camacho2000] Santos Costa, V.; Srinivasan, A.; and Camacho, R. 2000. A note on two simple transformations for improving the efficiency of an ILP system. In Cussens, J., and Frisch, A., eds., *Proceedings of the 10th International Conference on Inductive Logic Programming*, volume 1866 of *Lecture Notes in Artificial Intelligence*, 225–242. Springer-Verlag.

[Santos Costa1999] Santos Costa, V. 1999. Optimising bytecode emulation for Prolog. In *LNCS 1702, Proceedings of PPDP'99*, 261–267. Springer-Verlag.

[Sparrow1991] Sparrow, M. K. 1991. The application of network analysis to criminal intelligence: An assessment of the prospects. *Social Networks* 13:251–274.

[Srinivasan *et al.*1996] Srinivasan, A.; Muggleton, S. H.; Sternberg, M. J.; and King, R. D. 1996. Theories for mutagenicity: A study in first-order and feature-based induction. *Artificial Intelligence* 85:277–300.

[Srinivasan1999] Srinivasan, A. 1999. A study of two sampling methods for analysing large datasets with ILP. *Data Mining and Knowledge Discovery* 3(1):95–123.

[Srinivasan2001] Srinivasan, A. 2001. *The Aleph Manual*. URL: http://oldwww.comlab.ox.ac.uk/oucl/groups/machlearn/Aleph/aleph_toc.html.

[Tang, Mooney, & Melville2003] Tang, L. R.; Mooney, R. J.; and Melville, P. 2003. Scaling up ilp to large examples: Results on link discovery for counter-terrorism. In *submitted to the KDD-03 Workshop on Multi-Relational Data Mining*.

[Wasserman & Faust1994] Wasserman, S., and Faust, K. 1994. *Social Network Analysis: Methods & Applications*. Cambridge, UK: Cambridge University Press.

[Williams & Woessner1995a] Williams, P., and Woessner, P. N. 1995a. Nuclear material trafficking: An interim assessment. *Transnational Organized Crime* 1(2):206–238.

[Williams & Woessner1995b] Williams, P., and Woessner, P. N. 1995b. Nuclear material trafficking: An interim assessment, ridgway viewpoints. Technical Report 3, Ridgway Center, University of Pittsburgh.

[Williams2002] Williams, P. 2002. Patterns, indicators, and warnings in link analysis: The contract killings dataset. Technical Report 2878, Veridian Systems Division.

[Witten & Frank1999] Witten, I. H., and Frank, E. 1999. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. San Francisco: Morgan Kaufmann.

[Woessner1995] Woessner, P. N. 1995. Chronology of nuclear smuggling incidents: July 1991-may 1995. *Transnational Organized Crime* 1(2):288–329.

[Woessner1997] Woessner, P. N. 1997. Chronology of radioactive and nuclear materials smuggling incidents: July 1991-june 1997. *Transnational Organized Crime* 3(1):114–209.

[Wrobel2001] Wrobel, S. 2001. Inductive logic programming for knowledge discovery in databases. In Džeroski, S., and Lavrač, N., eds., *Relational Data Mining*. Berlin: Springer Verlag. 74–101.

[Zelezny, Srinivasan, & Page2002] Zelezny, F.; Srinivasan, A.; and Page, D. 2002. Lattice-search runtime distributions may be heavy-tailed. In *Proceedings of the 12th International Conference on Inductive Logic Programming*. Springer Verlag.

[Zelle & Mooney1996] Zelle, J. M., and Mooney, R. J. 1996. Learning to parse database queries using inductive logic programming. In *Proceedings of the 14th National Conference on Artificial Intelligence*, 1050–1055.