Establishing Identity Equivalence in Multi-Relational Domains

Jesse Davis, Inês Dutra, David Page, Vítor Santos Costa
Dep of Computer Science and Dep of Biostatistics and Medical Informatics
University of Wisconsin-Madison
{jdavis,dutra,page,vitor}@biostat.wisc.edu

Keywords: Information Extraction and Link Analysis, Knowledge Discovery and Dissemination

Abstract

Identity Equivalence or Alias Detection is an important topic in Intelligence Analysis. Often, terrorists will use multiple different identities to avoid detection. We apply machine learning to the task of determining Identity Equivalence. Two challenges exist in this domain. First, data can be spread across multiple tables. Second, we need to limit the number of false positives. We present a two step approach to combat these issues. First, we use Inductive Logic Programming to find a set of rules that are predictive of aliases. In the second step, we treat each learned rule as a random variable in a Bayesian Network. We use the Bayesian Network to assign a probability that two identities are aliases. We evaluate our technique on several data sets and find that layering Bayesian Network over the rules significantly increases the precision of our system.

1 Introduction

Determining *Identity Equivalence*, or Alias Detection, is a common problem in Intelligence Analysis. Two different identifiers are equivalent or *aliases* if both refer to the same object. One traditional example of aliasing centers around mistyped or variant author names in documents. For example, one might want to determine if a citation for V.S. Costa and one for Vítor S. Costa refer to the same author. In this situation one evaluates matches based on textual similarity. Furthermore, the central information comes from the surrounding text (Pasula et al. 2002). However, Intelligence Analysis involves more complex situations, and often syntactic similarity is either unavailable or inapplicable. Instead, aliases must be detected through object attributes and through interaction patterns.

The Intelligence Analysis domain offers two further challenges to this problem. First, information is commonly stored in relational database management systems (RDBMS) and involves multiple relational tables. The format of the data suggests using multi-relational datamining techniques such as Inductive Logic Programming (ILP). ILP systems learn *rules*, which can contain fields from different tables, in First Order Logic. Modern ILP systems can handle significant databases, containing millions of tuples.

A second challenge in Intelligence Analysis arises from *false positives*, or false alarms. In our task, a false positive corresponds to incorrectly hypothesizing that two names refer to the same individual. A false positive might have serious consequences, such as incorrectly adding individuals to a no-fly list. False positives will always cause valuable time to be wasted. False positives reduce trust in the system: if an expert system frequently gives spurious predictions, analysts will ignore its output. For all of these reasons, it is essential to limit the false positive rate. Unfortunately, intelligence analysis is particularly susceptible to false positives, as one is often trying to detect anomalies that occur rarely in the general population. For example, in a city with 1 million individuals, there are 499 billion possible alias pairs. In this case, even a false positive rate of only 0.001% will result in about 5 million false positives, or about five bad aliases per person.

We propose a two step methodology to address these challenges. First, we learn a set of rules that can achieve high recall, that is, they should be able to recognize most of the true aliases. Unfortunately, some of these rules may have poor precision, meaning that they falsely classify identity pairs as aliases. The second step addresses this problem. Instead of just considering each rule as an individual classifier, we treat each rule as a feature of a new classifier. We use machine learning methods to obtain a classifier that takes advantage of the characteristics of the individual rules. We use Bayesian Networks as our model, as they calculate the probability that a pair of identities are aliases.

We have evaluated our approach on synthetic datasets developed by Information Extraction & Transport, Inc. within the EAGLE Project (Schrag 2004). We were provided with artificial worlds, characterized by *individuals*, and relationships between these individuals. Our results show excellent performance for several of the datasets.

This paper is organized as follows. In Section 2 we discuss ILP applied to alias detection. In Section 3 we give a brief overview of Bayesian networks. In Section 4 we present and discuss our results. We compare our work with related work in Section 5. Finally, in Section 6 we provide a more in depth discussion of the datasets and our results.

2 ILP For Alias Detection

Inductive Logic Programming (ILP) is a framework for learning relational descriptions (Lavrac and Dzeroski 2001). Given sets of positive and negative examples and background knowledge, an ILP system learns a set of rules to discriminate between the positive and negative instances. ILP is appropriate for learning in multi-relational domains as the learned rules are not restricted to contain fields or attributes for a single table in a database.

We use Srinivasan's Aleph ILP System (Srinivasan 2001). Aleph uses the Progol algorithm (Muggleton 1995) to learn rules described as Prolog programs. Aleph induces rules in two steps. Initially, it selects an example and searches the databases for the facts known to be true about that specific example. The Progol algorithm is based on the insight that some of these facts should explain this example. If so, it should be possible to generalize those facts so that they would also explain the other examples. The algorithm thus generates generalized combinations of the facts, searching for the combinations with best performance.

One major advantage of using ILP is that it produces understandable results. We show a sample rule generated by Aleph:

alias(Id_1,Id_2) $\leftarrow$
suspect(Id_2),
suspect(Id_3),
phonecall(Id_2,Id_3),
phonecall(Id_3,Id_1).

The rule says that two individuals Id_1 and Id_2 may be aliases if **(i)** they both made phone calls to the same intermediate individual Id_3; and **(ii)** individuals Id_2 and Id_3 have the same attribute (suspect). The rule reflects that in this world model suspects are more likely to have aliases. Moreover, an individual and its aliases tend to talk to the same people.

The next rule uses different information:

alias(Id_1,Id_2) $\leftarrow$
has_capability(Id_1,Cap),
has_capability(Id_2,Cap),
group_member(Id_1,G),
group_member(Id_2,G),
isa(G,nonthreatgroup).

Two individuals may be aliases because they have a common capability, and because they both belong to the same non-threat group.

Clearly, these two rules are not precise as the patterns these rules represent could easily be applied to ordinary individuals. One observation is that we are only using the original database schema. An analyst might define views, or inferred relations, that highlight interesting properties of individuals. For instance, the first rule indicates that an individual and its aliases tend to communicate with the same people. We thus might want to compare sets of people an individual and its aliases talk to. In the spirit of aggregate construction for multi-relational learning (Knobbe et al. 2001; Neville et al. 2003; Perlich and Provost 2003), we have experimented with hand-crafting rules that use aggregates over properties commonly found in the ILP learned rules.

Even inventing new attributes, it is impossible to find a single rule that correctly identifies all aliases. In the next section, we discuss our approach for combining rules to form a better classifier.

3 Bayesian Networks

One of the drawbacks of applying ILP to this problem is that each database for a world is extremely large. The consequence is that it is intractable to use all the negative examples when learning the rules, which makes the final set of rules more susceptible to false positives. First, by sampling the negative examples, we have changed the proportion of the population that has aliases. Second, in ILP the final classifier traditionally consists of forming a disjunction over the learned clauses, resulting in a decision list. An unseen example is applied to each clause in succession until it matches one of them. If the example does not match any rule, then it receives the negative classification. Unfortunately, the disjunction of clauses maximizes the number of false positives. These issues suggest a simple approach where we represent each learned rule as an attribute in a classifier. We used Bayesian networks to combine the rules for two reasons. First, they allow us to set prior probabilities to reflect the true proportion of the population that has aliases. Second, each prediction has a probability attached to it. We can view the probability as a measure of confidence in the prediction. We experiment with several different Bayes net models for combining the rules. Naïve Bayes (Pompe and Kononenko 1995) is straightforward approach that is easy to understand and fast to train.

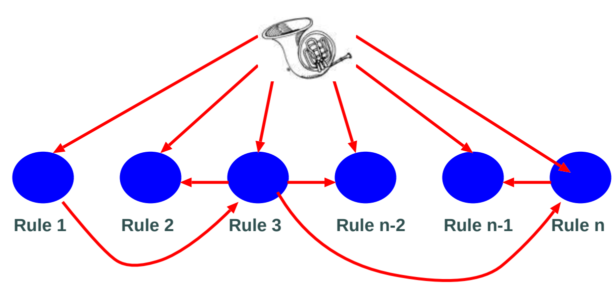

Figure 1: A TAN Bayes Net.

The major drawback with Naïve Bayes is that it assumes that the clauses are independent of each other given the class value. Often, we expect the learned rules to be strongly related. We used Tree Augmented Naïve Bayes networks (TAN) (Friedman et al. 1997), which allows for a slightly more expressive model. Figure 1 shows an example of a TAN network. Friedman, Geiger and Goldszmidt evaluated the algorithm for its viability on classification tasks. The TAN model, retains the basic structure of Naïve Bayes, but also permits each attribute to have at most one other parent, allowing the model to capture dependencies between attributes. To decide which arcs to include in the augmented network, the algorithm makes a complete graph between all the non-class attributes, where the weight of each edge is given as the conditional mutual information between those two attributes. A maximum weight spanning tree is constructed over this graph, and the edges that appear in the spanning tree are added to the network. Geiger proved that the TAN model can be constructed in polynomial time with a guarantee that the model maximizes the Log Likelihood of the network structure given the dataset (Friedman et al. 1997).

We have also experimented with other structure learning approaches, such as the Sparse Candidate algorithm (Friedman et al. 1999), but did not obtain significant improvements, as discussed by Davis et al. (2004).

4 Experiments

This section presents our results and analysis of the performance of our system on EAGLE datasets (Schrag 2004). The datasets are generated by simulating an artificial world with large numbers of relationships between agents. The data focuses on *individuals* which have a set of attributes, such as the capability to perform some actions. Individuals may also obtain resources, which might be necessary to perform actions. Individuals belong to groups, and groups participate in a wide range of *events*. In our case, given that some individuals may be known through different identifiers (e.g., through two different phone numbers), we were interested in recognizing whether two identifiers refer to the same individual.

The EAGLE datasets have evolved toward more complex and realistic worlds. We evaluate our system for datasets generated by two versions of the simulator. The results from the first version of the simulator are indexed with numbers while the newer datasets are indexed by roman numerals. Datasets vary with size, both in the number of individuals and in the activity level of each individual. Datasets also differ on observability, the amount of information available as evidence; on corruption, the number of errors; and on clutter, the amount of irrelevant information. Each dataset includes pre-processed data, called *primary* data, with group information, and on *secondary* data. The primary data contains a number of presumed aliases, which may or may not be true.

Each experiment was performed in two rounds. In the first round, the *dry-run*, we received a number of datasets plus their corresponding ground truth. This allowed us to experiment with our system and validate our methodology. In the second round, the *wet-run*, we received datasets without ground truth and were asked to present a hypothesis. We had a limited amount of time to do so. Later, we received the ground truth so that we could perform our own evaluation.

We adopted the following methodology. Rule learning is quite expensive in these large datasets. Moreover, we have found that most rules are relevant across the datasets, as we believe they capture aspects common to each simulated world. Consequently, we only performed rule learning during the dry-run. We used Srinivasan's Aleph ILP system (Srinivasan 2001) running on the YAP Prolog system. Ground-truth was used for training examples (and not used otherwise). The best rules from all datasets were passed forward to the wet-run.

We present results on the wet-run data in this paper. For the first set of data we used the wet-run datasets plus group information derived by the Kojak system (Adibi et al. 2004) (we did not have access to this information for the second batch of data). Using the rules learned from the training data, we converted each of the evaluation datasets into a set of propositional feature vectors, where each rule appears as an attribute in the feature vector. Each rule served as a binary attribute, which received a value of one if the rule matched the example and a zero otherwise.

We first report results from an earlier version of the EAGLE simulator, where only a single alias was allowed per entity. For space reasons, we only show results for three out of six of the datasets. Results for the other three are similar. We used five fold cross validation in these experiments.

For each application we show precision versus recall curves for the three methods: Naïve Bayes, TAN and voting. We used our own software for Naïve Bayes and TAN.

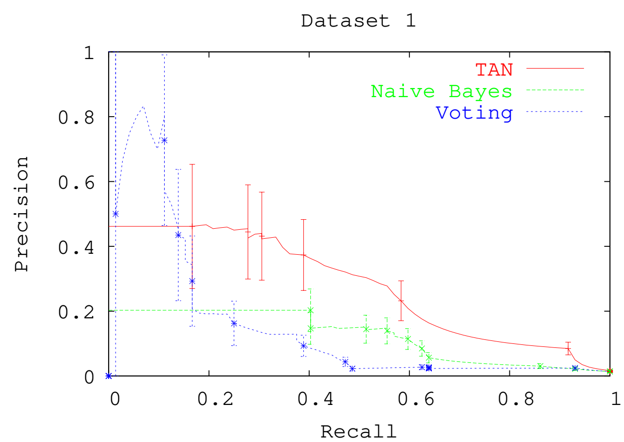

Figure 2: Precision Recall for Dataset 1

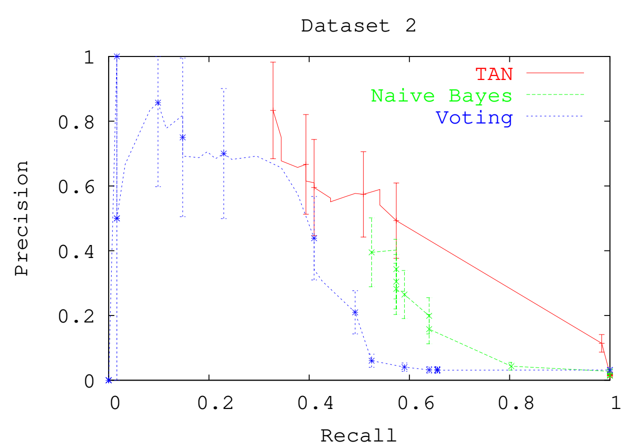

Figure 3: Precision Recall for Dataset 2

The Precision Recall curves for the different datasets are seen in Figures 2 through 4. We compare TAN and Naïve

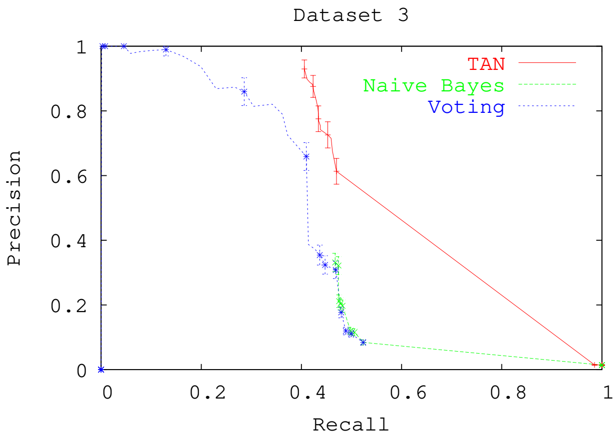

Figure 4: Precision Recall for Dataset 3

Bayes with unweighted voting, an ensemble method (Davis et al. 2004). On each curve, we included 95% confidence intervals on the precision score for selected levels of recall. The curves were obtained by averaging the precision and recall values for fixed thresholds. We achieved the best results for datasets 2 and 3, and did the worst on dataset 1, where precision levels did not achieve 0.5.

The second set of results comes from a more recent version of the simulator. Dataset sizes were at least as large, or bigger than before. The new simulator supported social network attributes, which could be used for the aliasing task. The error levels were increased and each individual could have up to six aliases. We used the same methodology as before, with two differences. First, we used ten fold cross validation in order to be able to perform significance tests. Second, we pooled the results across all ten folds to generate the precision recall curves. Due to space constraints, we only present results for 3 datasets: Datasets I, II, and III. The Precision/Recall curves are shown in Figures 5, 6, and 7.

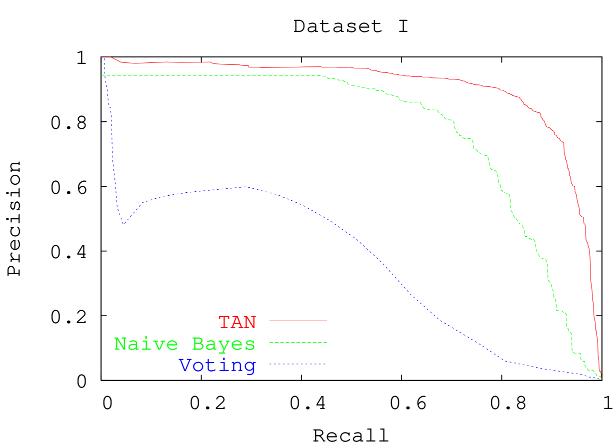

Figure 5: Precision Recall for Dataset I

Our results show much better performance for Datasets I and II. This is due to better rule quality. In this case we were able to find rules which have excellent recall, and the Bayes nets perform quite well at also achieving good precision. The results are particularly satisfactory using TAN on dataset I, as shown in Figure 5, where we can achieve precision over 60% for very high level recall. Dataset III was the hardest of all datasets for us. It shows a case where it is difficult to achieve both high precision and recall. This is because there is little information on individuals. In this case, improving recall requires trusting in only one or two rules, resulting in low precision.

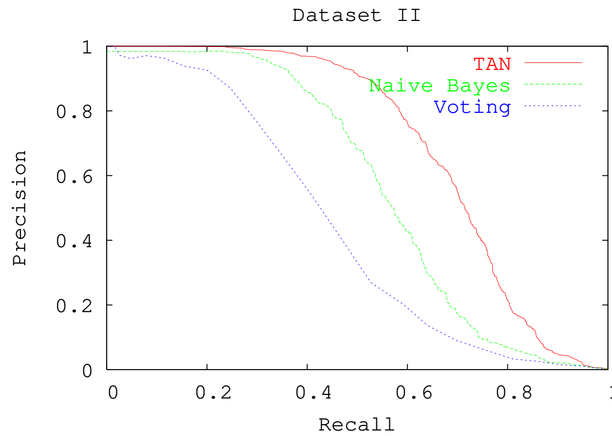

Figure 6: Precision Recall for Dataset II

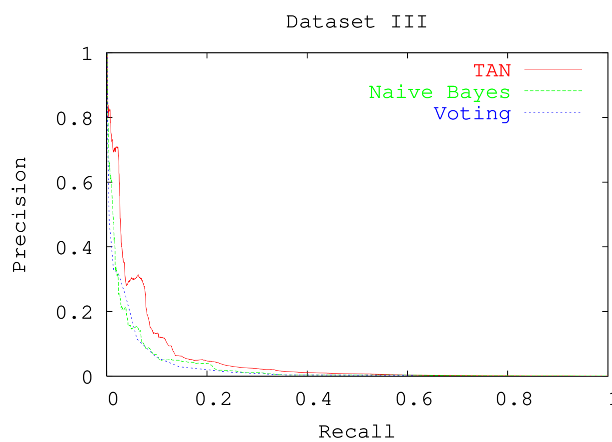

Figure 7: Precision Recall for Dataset III

The precision recall curve for the TAN algorithm dominates the curves for Naïve Bayes and ensemble voting on all six datasets. We have calculated the areas under the Precision/Recall curve for each fold in datasets I, II, III and the differences are statistically significant with 99% confidence. For each dataset, there are several places where TAN yields at least a 20 percentage point increase in precision, for the same level of recall over both Naïve Bayes and voting. On two of the six datasets, Naïve Bayes beats voting, while on the remaining four they have comparable performance. One reason for TAN's dominance compared to Naïve Bayes is the presence of rules which are simply refinements of other rules. The TAN model is able to capture some of these interdependencies, whereas Naïve Bayes explicitly assumes that these dependencies do not exist. The Naïve Bayes independence assumption accounts for the similar performance compared to voting on several of the datasets.

In situations with imprecise rules and a preponderance of negative examples, such as link discovery domains, Bayesian models and especially TAN provide an advantage. One area where both TAN and Naïve Bayes excel is in handling imprecise rules. The Bayes nets effectively weight the precision of each rule either individually or based on the outcome of another rule in the case of TAN. The Bayesian nets further combine these probabilities to make a prediction of the final classification allowing them to discount the influence of spurious rules in the classification process. Ensemble voting does not have this flexibility and consequently lacks robustness to imprecise rules. Another area where TAN provides an advantage is when multiple imprecise rules provide significant overlapping coverage on positive examples and a low level of overlapping coverage on negative examples. The TAN network can model this scenario and weed out the false positives. One potential disadvantage to the Bayesian approach is that it could be overly cautious about classifying something as a positive. The high number of negative examples relative to the number of positive examples, and the corresponding concern of a high false positive rate, helps mitigate this potential problem. In fact, at similar levels of recall, TAN has a lower false positive rate than voting.

5 Related Work

Identity Uncertainty is a difficult problem that arises in areas such as Citation Matching, Record Linkage and De-Duplication in Databases, Natural Language Processing, in addition to Intelligence Analysis. A seminal work in this area is the theory of Record Linkage (Fellegi and Sunter 1969), based on scoring the distances between two feature vectors (using Naïve Bayes in the original work) and merging records below some threshold. Systems such as Citeseer (Lawrence et al. 1999) apply similar ideas by using text similarity. The field of record matching has received significant contributions (Monge and Elkan 1996; Cohen and Richman 2002; Buechi et al. 2003; Bilenko and Mooney 2003; Zelenko et al. 2003; Hsiung et al. 2004). On the other hand, it has been observed that interactions between identifiers can be crucial in identifying them (Morton 2000). Pasula et al. (2002) use relational probabilistic models to establish a probabilistic network of individuals, and then use Markov Chain Monte Carlo to do inference on the citation domain. McCallum and Wellner (2003) use discriminative models, Conditional Random Fields, for the same task. These approaches rely on prior understanding of the features of interest, usually text based. Such knowledge may not be available for Intelligence Analysis tasks.

Detecting features of interest was therefore our first step, and the present work fits into the popular category of using ILP for feature construction. Such work treats ILP-constructed rules as Boolean features, re-represents each example as a feature vector, and then uses a feature-vector learner to produce a final classifier. To our knowledge, the work closest to ours is the one by Pompe and Kononenko (1995), who were the first to apply Naïve Bayes to combine clauses. Other work in this category was by Srinivasan and King (1997), for the task of predicting biological activities of molecules from their atom-and-bond structures. Some other work, especially on propositionalization of First Order Logic (FOL) (Alphonse and Rouveirol 2000), has been developed that converts the training sets to propositions and then applies feature vector techniques to the converted data. This is similar to what we do, however we first learn from FOL and, then learn the network structure and parameters using the feature vectors obtained with the FOL training, resulting in much smaller feature vectors than in other work.

Our paper contributes two novel points to this category of work. First, it highlights the relationship between this category of work and ensembles in ILP, because when the feature-vector learner is Naïve Bayes the learned model can be considered a weighted vote of the rules. Second, it shows that when the features are ILP-learned rules, the independence assumption in Naïve Bayes may be violated badly enough to yield a high false positive rate. This false positive rate can be reduced by permitting strong dependencies to be explicitly noted, through learning a Tree Augmented Naïve Bayes network (TAN).

6 Conclusions and Future Work

Identity Equivalence is an important problem in Intelligence Analysis. Quite often, individuals want to hide their identities, and therefore we cannot rely on textual information. Instead, we need to use attributes and contextual information. We show that good results can be achieved by using multi-relational learning to learn rules, whose output is then combined to lower the false positive rate. We were particularly interested in Bayesian methods for the latter because they associate a probability with each prediction, which can be thought of as the classifier's confidence in the final classification. We compare how three different approaches for combining rules learned by an ILP system perform on an application where data is subject to corruption and unobservability. We demonstrate experimentally that we can significantly lower the false positive rate through rule combination schemes.

We obtained the best precision recall results in our application using a TAN network to combine rules. Precision was a major concern to us due to the high ratio of negative examples to positive examples. TAN had better precision than Naïve Bayes or unweighted voting, because it is more robust at handling redundancy between rules.

In future work we plan to experiment with different applications and Bayesian network structures. We are interested in learning rules with aggregates. We plan to further continue work based on the observation that we learn a single CLP($\mathcal{BN}$) network (Santos Costa et al. 2003). This observation suggests that a stronger coupling between the learning phases could be useful.

7 Acknowledgments

Support for this research was partially provided by U.S. Air Force grant F30602-01-2-0571. We would also like to thank Irene Ong, Bob Schrag, and Jude Shavlik for all their help. Inês Dutra and Vítor Santos Costa are on leave from Federal University of Rio de Janeiro, Brazil.

References

J. Adibi, H. Chalupsky, E. Melz, and A. Valente. The KOJAK Group Finder: Connecting the Dots via Integrated Knowledge-Based and Statistical Reasoning. In *Proceedings of the Sixteenth Innovative Applications of Artificial Intelligence Conference (IAAI-04)*, page To Appear, 2004.

E. Alphonse and C. Rouveirol. Lazy propositionalisation for relational learning. In Horn W., editor, *14th European Conference on Artificial Intelligence, (ECAI'00) Berlin, Allemagne*, pages 256–260. IOS Press, 2000.

Mikhail Bilenko and Raymond J. Mooney. Adaptive duplicate detection using learnable string similarity measures. In *KDD*, pages 39–48, 2003.

Martin Buechi, Andrew Borthwick, Adam Winkel, and Arthur Goldberg. Cluemaker: A language for approximate record matching. In *IQ*, pages 207–223, 2003.

William W. Cohen and Jacob Richman. Learning to match and cluster large high-dimensional data sets for data integration. In *KDD*, pages 475–480, 2002.

Jesse Davis, Vítor Santos Costa, Irene M. Ong, David Page, and Ins C. Dutra. Using Bayesian Classifiers to Combine Rules. In *3rd Workshop on Multi-Relational Data Mining*, Seattle, USA, August 2004.

I. Fellegi and A. Sunter. Theory of record linkage. *Journal of the American Stastistical Association*, 64:1183–1210, 1969.

N. Friedman, I. Nachman, and D. Pe'er. Learning bayesian network structure from massive datasets: The "sparse candidate" algorithm. In *Proceedings of the 15th Annual Conference on Uncertainty in Artificial Intelligence (UAI-99)*, pages 206–215, San Francisco, CA, 1999. Morgan Kaufmann Publishers.

Nir Friedman, David Geiger, and Moises Goldszmidt. Bayesian networks classifiers. *Machine Learning*, 29:131–163, 1997.

Paul Hsiung, Andrew Moore, Daniel Neill, and Jeff Schneider. Alias detection in link data sets. Master's thesis, Carnegie Mellon University, March 2004. Carnegie Mellon University.

Arno J. Knobbe, Marc de Haas, and Arno Siebes. Propositionalisation and aggregates. In *PKDD01*, pages 277–288, 2001.

N. Lavrac and S. Dzeroski, editors. *Relational Data Mining*. Springer-Verlag, Berlin, September 2001. ISBN 3-540-42289-7.

Steve Lawrence, C. Lee Giles, and Kurt D. Bollacker. Autonomous citation matching. In *Agents*, pages 392–393, 1999.

Andrew McCallum and Ben Wellner. Toward conditional models of identity uncertainty with application to proper noun coreference. In *IIWeb*, pages 79–84, 2003.

Alvaro E. Monge and Charles Elkan. The field matching problem: Algorithms and applications. In *KDD*, pages 267–270, 1996.

Thomas S. Morton. Coreference for NLP Applications. In *ACL*, 2000.

S. Muggleton. Inverse entailment and Progol. *New Generation Computing*, 13:245–286, 1995.

Jennifer Neville, David Jensen, Lisa Friedland, and Michael Hay. Learning relational probability trees. In *KDD '03*, pages 625–630. ACM Press, 2003. ISBN 1-58113-737-0.

Hanna Pasula, Bhaskara Marthi, Brian Milch, Stuart J. Russell, and Ilya Shpitser. Identity uncertainty and citation matching. In *NIPS*, pages 1401–1408, 2002.

Claudia Perlich and Foster Provost. Aggregation-based feature invention and relational concept classes. In *KDD '03*, pages 167–176, 2003. ISBN 1-58113-737-0.

U. Pompe and I. Kononenko. Naive Bayesian classifier within ILP-R. In L. De Raedt, editor, *Proceedings of the 5th International Workshop on Inductive Logic Programming*, pages 417–436. Department of Computer Science, Katholieke Universiteit Leuven, 1995.

Vítor Santos Costa, David Page, Maleeha Qazi, and James Cussens. CLP($\mathcal{BN}$): Constraint Logic Programming for Probabilistic Knowledge. In *Proceedings of the 19th Conference on Uncertainty in Artificial Intelligence (UAI03)*, pages 517–524, Acapulco, Mexico, August 2003.

Robert C. Schrag. EAGLE Y2.5 Performance Evaluation Laboratory (PE Lab) Documentation Version 1.5. Internal report, Information Extraction & Transport Inc., April 2004.

A. Srinivasan. *The Aleph Manual*, 2001.

A. Srinivasan and R. King. Feature construction with inductive logic programming: A study of quantitative predictions of biological activity aided by structural attributes. In S. Muggleton, editor, *Proceedings of the Sixth Inductive Logic Programming Workshop*, LNAI 1314, pages 89–104, Berlin, 1997. Springer-Verlag.

Dmitry Zelenko, Chinatsu Aone, and Anthony Richardella. Kernel methods for relation extraction. *Journal of Machine Learning Research*, 3:1083–1106, 2003.