Case-Based Collective Classification

Luke K. McDowell[1], Kalyan Moy Gupta[2], and David W. Aha[3]

[1]Dept. of Computer Science; U.S. Naval Academy; Annapolis, MD 21402
[2]Knexus Research Corp.; Springfield, VA 22153
[3]Navy Center for Applied Research in Artificial Intelligence;
Naval Research Laboratory (Code 5515); Washington, DC 20375
lmcdowel@usna.edu, kalyan.gupta@knexusresearch.com, david.aha@nrl.navy.mil

Abstract

This is the first paper on textual case-based reasoning to employ collective classification, a methodology for simultaneously classifying related cases that has consistently attained higher accuracies than standard classification approaches when cases are related. Thus far, case-based classifiers have not been examined for their use in collective classification. We introduce *Case-Based Collective Classification* (CBCC) and report that it outperforms a traditional case-based classifier on three tasks. We also address issues of case representation and feature weight learning for CBCC. In particular, we describe a cross-validation approach for tuning feature weights and show that it increases CBCC accuracy on these tasks.

Introduction

Classification is the task of assigning one or more class labels to an unlabeled instance (or *case*). Many methods for automatically inducing classifiers exist (e.g., those that induce Bayesian networks, decision trees, neural networks, or rules). Case-based classification (CBC) is one such method: it classifies an unlabeled case by retrieving closely matching labeled cases and reusing their labels.

An underlying assumption of traditional classification methods is that the cases are independent of each other. For example, this has been assumed in most previous research on case-based classification (Aha and Marling 2005). However, there are many classification tasks where cases are implicitly or explicitly related. One such task is that of assigning classes (topics) to web pages (cases), which may contain hyperlinks to each other. Hyperlinked web pages are more likely to share common class labels than non-linked pages, and this is an important factor to consider when classifying them. Such *auto-correlation* (correlation of class labels among interrelated cases) has been observed in a wide variety of data (Jensen and Neville 2002) including data sets where the relationships are implicit. For example, email messages between two people are likely to share topics.

Collective classification is a methodology that simultaneously classifies cases which may be interrelated. Recent research has shown that it can significantly increase classification accuracies over standard classification methods when cases are interrelated (Taskar, Abbeel, and Koller 2002; Lu and Getoor 2003; Neville and Jensen 2003). Classifiers that have been used in collective classification research include naïve Bayes, logistic regression, dependency networks, and Markov networks. CBC has not been used for collective classification.

In this paper, we introduce and develop a *Case-Based Collective Classification* (CBCC) methodology. We demonstrate that CBCC yields higher classification accuracy than standard CBC approaches for three textual CBC tasks. To do so, we create an appropriate relational representation for cases. We also describe a cross-validation method for learning feature weights and show that it significantly increases CBCC accuracy.

We next describe collective classification, discuss related work, and introduce CBCC. Our empirical evaluation and discussion of future work follows.

Collective Classification

In some classification tasks, the unlabeled cases may be implicitly or explicitly related. For example, consider classifying a university web page as belonging to a faculty member or a student. Many of the student and faculty web pages may hyperlink to each other. *Standard classifiers* typically ignore such relations or links and would independently classify a web page by considering only the features derived from its contents (e.g., words it contains). Classification accuracy can be increased by carefully adding features derived from the content of related cases (e.g., words from the hyperlinked web pages) (Yang, Slattery, and Ghani 2002). Even greater accuracy increases can occur when the class label(s) of the related web pages are used to derive relevant *relational* features for classification (Jensen, Neville, and Gallagher 2004). However, some or all of the class labels of the related web pages are initially unknown and need to be *estimated* or *inferred* to bootstrap the classification process. This can be done in several ways. For instance, initial estimates of class labels can be obtained using content features only. Next, these estimates could be used to compute the relational features' values and reclassify the cases. This

Table 1. A partial summary of research on collective classification

Publication	Base Classifier	Inference Method	Datasets	Increase in % acc.vs. non-collective
Chakrabarti *et al*. 1998	Naïve Bayes	Relaxation Labeling	Patent DB, Yahoo	15-47
Neville and Jensen 2000	Naïve Bayes	Iterative (gradual commit)	SEC (corporations)	6-12
Taskar *et al*. 2002	Markov Network	Belief Propagation	WebKB	2-10
Lu and Getoor 2003	Logistic Regression	Iterative (ICA)	Cora, CiteSeer, WebKB	2-8
Neville and Jensen 2003	Dependency Network	Gibbs Sampling	IMDb, Cora	10-47
Jensen *et al*. 2004	Naïve Bayes	Gibbs Sampling	Yeast	11
This paper	Case-Based (k-NN)	Iterative (ICA)	Cora, CiteSeer, WebKB	5-13

process of inferring and reclassifying could be repeated until the class labels converge.

Such an iterative approach for simultaneously classifying interrelated cases using estimated or inferred relational features is called *collective classification* (Jensen *et al*. 2004). These approaches have three characteristics:

- *Related cases*: The cases to be classified (e.g., web pages, emails) are explicitly or implicitly inter-related.
- *Collective inference*: A collective inference algorithm is used to update the class labels (or class conditional probabilities), which are subsequently used to identify the values of relational features. Several inferencing schemes have been used for collective classification (see Table 1). In the *Iterative Classification Algorithm* (ICA) (Lu and Getoor 2003), class labels are initially assigned based purely on non-relational features or a priori knowledge of the class distribution. Subsequently, they are updated in each iteration by the classifier. Neville and Jensen (2000) described a "gradual commitment" version of this iterative approach in which no initial label assignments are made. Instead, only a predefined proportion of the unlabeled cases are labeled per iteration, starting with the cases whose assignment confidence is highest. *Relaxation labeling* is yet another approach (Chakrabarti, Dom, and Indyk 1998), some forms of which can be viewed as essentially the same as ICA where class labels are assigned using estimated class conditional probabilities for each linked case (Sen and Getoor 2006). *Belief propagation* and *Gibbs Sampling* have also been used to estimate the joint probabilities of class labels. We use ICA in this paper. This algorithm was the most reliable performer on a range of synthetic data (Sen and Getoor, 2006), and it and its variants have also been effective on publicly available (non-synthetic) data sets (Neville and Jensen 2000; Lu and Getoor, 2003).
- *Classifier*: The classifier uses inferred *relational features* in addition to *non-relational features* to classify the set of related cases. For example, the words in a webpage can be the non-relational features and the most common class label across the other pages hyperlinked to that page can serve as an inferred relational feature. Naïve Bayes, Markov networks, logistic regression, and dependency networks have been used for this purpose (see Table 1). To our knowledge, no previous collective classification approach has used a case-based classifier.

Case-Based Collective Classification

We use the k-nearest neighbor (k-NN) rule for case-base classification. k-NN requires defining the case representation and the similarity function, which may employ algorithms for feature selection and/or weighting. In addition, we must select a collective inference scheme. Below we describe our case representation and the cross-validation procedure that we use to learn feature weights for the similarity function. In addition, we detail our collective inference scheme.

Case Representation

We focus on collective classification tasks involving data sets that are predominantly textual (e.g., classifying web pages or research publications). Our case representation for collective classification includes the following:

- *Non-relational (content) features*: We use a bag-of-words representation for the textual content of cases (e.g., Gupta, Aha, and Moore 2006). In particular, we use a binary representation where, for the selected set of features, the feature corresponding to a word is assigned the value *true* if the word occurs in the case and is assigned *false* otherwise.
- *Relational features*: We include several binary features whose values are computed by applying a threshold on the number of links to other matching labeled training cases, and also, during testing, to test cases (whose labels are uncertain). For example, a relational feature f_{B2} is *true* for a case i if i has at least two hyperlinked cases that currently have class B as their label. While other representations for relational features are possible, in this introductory paper we decided to use only these binary features. This decision was guided in part by our empirical observations that this approach compensates for disparities between the average number of links per case in the training set vs. in the test set, a disparity that can occur when the test set links to the training set.

```
CBCC(Tr,Te,NR,R,n) =
// Tr=Training data, Te=Test data, NR=non-relational features,
// R=relational features, n=# iterations, W=feature weights
1  Tr.R.values←setRelFeatures(Tr,R)          // Initialize
2  W←learn_feature_weights(Tr,NR,R)          // Train
3  Te.ClassLabels←classify(Te,Tr,W,NR,∅)     // Bootstrap
4  for i =1 to n                             // Iterate
5     Te.R.values←setRelFeatures(Te∪Tr,R)    // Update
6     Te.ClassLabels←classify(Te, Tr,W,NR,R) // Classify
7  return Te.ClassLabels
```

Figure 1. Generic case-based collective classification pseudocode, where the classifier is a weighted k-NN rule

Similarity Function

We selected a weighted overlap similarity function for our k-NN classifier. In particular, for cases *i* and *j*:

$$sim(i, j) = \sum_k w_k (1 - |f_k(i) - f_k(j)|) / \sum_k w_k$$

where w_k is the weight of feature *k*, and $f_k(i)$ is the (binary) value of feature *k* for case *i*. Weighted similarity is used for voting. We examined three weight-learning techniques:

- *Equal weights* (**EW**): This assigns the same weight to all features.
- *Information gain weights* (**IGW**): Information gain is a popular and effective method for feature weighting in textual case-based reasoning (Gupta *et al.* 2006; Wiratunga, Koychev, and Massie 2004). However, its effectiveness in CBCC has not been evaluated and is potentially problematic because the intermediate values of relational features (i.e., that depend on the unknown labels of linked test cases) during testing may be incorrect. Hence, weights learned on (fully correct) training data may be misleading.
- *Cross-validation weights* (**CVW**): To address this problem with IGW, we devised a cross-validation method for learning feature weight settings. In this method, values for non-relational feature weights are, like IGW, set using information gain. However, the weight values of all relational features are held equal, and this one value is linearly searched by estimating its resultant classification accuracy on a hold-out set. We expect that the weight settings for relational features, whose values will be uncertain and contain some noise, will be lower than those set using IGW, and that CVW will subsequently yield higher classification accuracies.

Collective Inference

Figure 1 shows CBCC, which is an adaptation of Lu and Getoor's (2003) Iterative Classification Algorithm for use with CBC. After initializing the relational feature values for the training set (step 1), training involves assigning feature weight settings using one of the three learning algorithms (step 2). Next, a bootstrap step (3) makes initial predictions for the test cases using only non-relational features. Then it repeatedly updates the relational features' values based on these predicted classifications and reclassifies the test set (using all features), continuing until a stopping criterion is satisfied (steps 4-6). After empirical analyses, we decided to use a fixed number of iterations (n=10, which Neville and Jensen (2000) also used).

Table 2. Data sets summary

Characteristics	Cora	CiteSeer	WebKB
Cases	2708	3312	1541
Average links per case	4.01	2.77	6.59
Classes	7	6	6
Non-rel. features available	1433	3703	100
Non-rel. features used	100	100	100
Relational features used	21	18	36
Folds	3	3	4

Evaluation

We evaluate hypotheses corresponding to the following two claims:

1. CBCC attains higher accuracies than standard CBC on data with relational information.
2. The CVW is the most effective weighting technique, among the three described here, for CBCC.

Data Sets

We tested these claims on three data sets (see Table 2) that were also used in previous studies (see Table 1):

1. ***Cora*** (McCallum *et al*. 2000): A collection of machine learning publications categorized into seven classes.
2. ***CiteSeer*** (Lu and Getoor 2003): A collection of research publications drawn from CiteSeer (2006).
3. ***WebKB*** (Craven *et al*. 1998): This contains web pages from four computer science departments categorized into six classes (*Faculty, Student, Staff, Course, ResearchProject,* or *Other*). *Other* is problematic because it is too general, representing 74% of the pages. Like Taskar et al. (2002), we discarded all *Other* pages that did not have at least three outgoing links, yielding a total of 1541 cases of which 30% are *Other*.

For Cora and CiteSeer, links exist between the test and training sets that can assist the collective inference process. This is the "in-sample" classification task described by Neville and Jensen (2005). For WebKB, there are no links between the different schools, so we measure the accuracy on the "out-of-sample" classification task.

Case Representation. For WebKB, we used the 100 most frequent words (all that were available in our version of the dataset) as the non-relational features. For Cora and CiteSeer, we instead used information gain on the training set to identify and select the 100 highest-scoring words as the non-relational features. Using more words provided little improvement in baseline classification accuracy, and thus we chose 100 words for consistency with WebKB.

When developing relational features, we considered the directionality of links. We empirically established that undirected links perform better than directed links for Cora and CiteSeer, and directed links work best for WebKB. This is consistent with our intuition that citation links in Cora and CiteSeer are semantically fairly symmetric, while the WebKB hyperlinks are not. Therefore, we represented WebKB's relational features using directed links and used undirected links to represent relational features for Cora and CiteSeer. For each data set, we created relational features with thresholds of 1, 2, and 3 for each class in the dataset (e.g., corresponding to Cora's 7 classes, we created 21 relational binary features). WebKB has six classes, but has 36 relational features due to its use of directed links.

Weight learning: We applied CVW as follows. For each training set, the weights of non-relational features were set using information gain. Two of the training set folds were then selected. We ran the classification algorithm, training with the first fold and evaluating accuracy on the second (hold-out) set. We repeated this process, each time setting the relational feature weights to one of 13 settings ranging from 0.01 to 200.0. The weight yielding the best accuracy was then used for evaluation on the test set.

Implemented Algorithms

We tested three algorithms in our experiments:

1. `NonColl`: This version implements the standard CBC that uses only the non-relational features.
2. $\texttt{Coll}_0$: This version operates on cases represented with non-relational and relational features. However, it performs only a single classification step, where each test set relational feature is used only if its value can be determined using just the links to the training set. $\texttt{Coll}_0$ is intended to measure the performance improvement due solely to adding relational features, even without performing iterative collective inference.
3. $\texttt{Coll}_n$: This version implements CBCC when iterating n times. We set n=10 in our experiments.

Finally, we empirically set k=11 for this evaluation.

Performance Measure

We compared all the algorithms for their average classification accuracy, which is the proportion of cases from the test sets that are correctly classified.

Hypotheses

1. CBCC ($\texttt{Coll}_n$) outperforms standard CBC with ($\texttt{Coll}_0$) and without (`NonColl`) relational features.
2. CVW yields significantly higher accuracies than EW and IGW for CBCC ($\texttt{Coll}_n$).

Test Procedure

We conducted an n-fold cross-validation study for each algorithm. The Cora and CiteSeer data sets, as provided to us, were split into three roughly equal-sized folds (intended to preserve linking within a fold), which we used as is. We did not use randomly generated folds because that could remove the naturally occurring relations, resulting in folds that would be unrealistic for a collective classification task. For WebKB, we conducted a 4-fold cross-validation study, treating each of the four schools as a separate fold.

Analysis

We analyzed the results for each data set and performed a joint analysis by pooling the observations from all the data sets. Our conclusions are based on a one-tailed paired t-test for statistical significance at the 95% level.

Results

Table 3 displays the classification accuracies averaged over all the folds for each algorithm. The best performance for each algorithm for each dataset is shown in bold. We first evaluate Hypothesis 1 for each weighting method.

EW. For Cora, $\texttt{Coll}_n$'s accuracy is significantly higher than `NonColl` and $\texttt{Coll}_0$ (71.2 vs. 59.6 [p=0.005] and 71.2 vs. 64.9 [p=0.003]). For CiteSeer, $\texttt{Coll}_n$ does *not* significantly outperform `NonColl` and $\texttt{Coll}_0$ (65.0 vs. 59.2 [p=0.067] and 65.0 vs. 59.7 [p=0.076]). For WebKB, $\texttt{Coll}_n$'s accuracies are again significantly higher than `NonColl` and $\texttt{Coll}_0$ (60.2 vs. 50.9 [p=0.004] for both).

Using a pooled data analysis, $\texttt{Coll}_n$'s accuracies are significantly higher than `NonColl` (64.94 vs. 56.00 [p=0.000]) and $\texttt{Coll}_0$ (64.94 vs. 57.78 [p=0.000]).

Comparing the performance of $\texttt{Coll}_0$ with `NonColl`, we found that adding relational features alone, even without collective inference, can increase classification accuracy (e.g., 5.3% in Cora and 0.5% in CiteSeer). However, there was no increase for WebKB because it has no hyperlinks from the test cases to the training cases. That is, all relational features with $\texttt{Coll}_0$ are set to `false`.

To assess the magnitude of improvement attributable to collective inference, we compared the classification accuracies of $\texttt{Coll}_n$ with $\texttt{Coll}_0$. The increases (ranging between 5.3% and 9.3%) are substantial.

IGW. For Cora, $\texttt{Coll}_n$ does *not* significantly increase accuracies compared with `NonColl` (67.4 vs. 62.4 [p=0.062]) and $\texttt{Coll}_0$ (67.4 vs. 65.6 [p=0.225]). However, for WebKB, $\texttt{Coll}_n$ does significantly outperform `NonColl` and $\texttt{Coll}_0$ (70.5 vs. 59.5 [p=0.001] for both). The performance on CiteSeer is inconsistent with that of Cora and WebKB. Although statistically not significant, on Citeseer $\texttt{Coll}_n$ performs poorly compared to `NonColl` and $\texttt{Coll}_0$ (53.3 vs. 62.1 [p=0.226] and 53.3 vs. 62.5 [p= 0.218]).

As anticipated, IGW with $\texttt{Coll}_n$ had an unpredictable and sometimes adverse effect on accuracy. Other experiments confirmed this even with fewer relational features (e.g., only features with a threshold of 1). However, as discussed later, CVW resolves this problem. Thus, we do not perform a pooled analysis for IGW.

CVW. For Cora, $\texttt{Coll}_n$ attained significantly higher accuracies than `NonColl` (75.4 vs. 62.4 [p=0.002]).

Table 3. Average % classification accuracies (generalization)

Algorithm	Cora			CiteSeer			WebKB		
	EW	IGW	CVW	EW	IGW	CVW	EW	IGW	CVW
NonColl	59.6	**62.4**	**62.4**	59.2	**62.1**	**62.1**	50.9	**59.5**	**59.5**
Coll_0	64.9	65.6	**71.8**	59.7	62.5	**63.2**	50.9	**59.5**	**59.5**
Coll_n	71.2	67.4	**75.4**	65.0	53.3	**66.9**	60.2	**70.5**	69.8

However, it does *not* significantly outperform Coll_0 (75.4 vs. 71.8 [p=0.089]). For CiteSeer, Coll_n does not significantly outperform NonColl (66.9 vs. 62.1 [p=0.051]) but it does significantly outperform Coll_0 (66.9 vs. 63.2 [p=0.038]). For WebKB, Coll_n significantly outperforms NonColl and Coll_0 (69.8 vs. 59.5 [p=0.000] for both).

Using a pooled data analysis, we found that, for CVW, Coll_n yields significantly higher accuracies than NonColl (70.60 vs. 61.13 [p=0.000]) *and* Coll_0 (70.60 vs. 64.31 [p=0.000]).

The magnitude of performance improvements of Coll_n versus Coll_0 on CVW are 3.6% for Cora, 3.7% for CiteSeer, and 10.3% for WebKB.

***Overall conclusion*:** Based on the pooled analysis, we accept Hypothesis 1 for EW and CVW, but not for IGW.

Hypothesis 2. A per dataset analysis shows that CVW does significantly increase classification accuracy on Cora (e.g., 75.4 vs. 71.2 [p=0.031] for EW, and 75.4 vs. 67.4 [p=0.041] for IGW). Although similar improvements occurred for CiteSeer, they are not significant (66.9 vs. 65.0 [p=0.072]) and (66.9 vs. 53.3 [p=0.115]). For WebKB, we found that CVW significantly outperforms EW (69.8 vs. 60.2, [p=0.001]) but slightly under performs IGW, although this difference is not significant (69.8 vs. 70.5 [p=0.297]).

Pooling our observations across all the datasets to assess the overall effectiveness of CVW, we found that it does provide a significant advantage over EW (70.6 vs. 64.9 [p=0.000]) and IGW (70.6 vs. 64.4 [p=0.033]).

Overall Conclusion: Based on the pooled analysis, we accept Hypothesis 2.

We posited earlier that IGW, when used with collective inference, can adversely impact classification accuracy by amplifying the errors from incorrectly predicted relational feature values. We expected CVW to counter this adverse effect by selecting a smaller weight setting for the relational features when compared with those selected by IGW. This occurred for Cora and CiteSeer (see Table 4). The average total weight of relational features selected by CVW is substantially lower than those selected by IGW (e.g., 1.12 vs. 5.01 for Cora). However, CVW's search procedure found *higher* weights for WebKB (3.60 vs. 1.40). This explains its marginally lower classification performance on WebKB. This inability of CVW to locate better weight settings could be attributed to the nature of hyperlinks or the smaller size of the WebKB training set. Further investigation is needed to confirm this conjecture.

Table 4. Average total weight of relational features

Data Set	IGW	CVW
Cora	5.01	1.12
CiteSeer	2.79	0.66
WebKB	1.40	3.60

Table 3 shows that CVW yields the best classification accuracies with Coll_0, when relations from test to training sets are available. For Cora, the improvement versus EW is 6.9% and 6.2% versus IGW. Similar (though smaller) improvements occurred with CiteSeer.

Overall, CVW improves performance in part because it is a wrapper approach (Kohavi and John 1997); it directly uses classification performance to select the best weights. In contrast, IGW uses a filter approach in which IG is a proxy measure instead of the classifier's accuracy. In general, wrapper approaches often yield higher accuracies.

Discussion

Previous research on case-based classification (e.g., Gupta *et al*., 2006), relational case representations (e.g., Bergmann, Kolodner, and Plaza 2005), and textual case-based reasoning (Weber, Ashley, and Brüninghaus 2005) has not examined using inter-case relations to classify all cases simultaneously. In contrast, CBCC's context is that the cases are related and knowing the label of one can affect the prediction for another, related case.

For the datasets we examined, relational features appear to be informative and suitable for improving performance. Where links between training and test cases are available (Cora, CiteSeer), adding relational features, even without collective inference, increased accuracy. However, the best performance for all the datasets was achieved using CBCC. The magnitude of CBCC's improvements is similar to what Lu and Getoor (2003) reported for ICA with logistic regression on Cora and CiteSeer. However, for WebKB we found a much larger performance improvement than they did; we attribute this to their different data pre-processing which removed *all* pages in the *Other* category (we removed just some), resulting in a less connected graph.

Because relational data is common and important in many applications, CBCC appears to be a useful new technology, particularly when the data meets the following conditions. First, the data must contain (or be amenable to inferring) a sufficient number of inter-case relations. The number of such relations necessary will vary; our datasets contained between 2.8 and 6.6 relations per case. Second, knowing the class of one case must be informative in

predicting the class of a linked case; auto-correlation is a common occurrence that meets this requirement (Jensen and Neville 2002). Finally, we found that CBCC improved classification accuracies for baseline accuracies as low as 51% (obtained for WebKB). There may be a lower bound on baseline accuracy below which CBCC may not be effective, which we will analyze in our future research.

Conclusion

We introduced case-based collective classification (CBCC) and demonstrated that it can significantly increase classification accuracy for relational data. We summarized collective classification and explained how to apply such techniques with a k-nearest neighbor classifier. Our choice of relational features led to good performances for three classification tasks. We also explored the impact of feature weighting. CBCC accuracy generally increased when using feature weighting, although information gain often performed poorly when the relational features' values were incorrectly predicted. In response, we introduced a cross-validation feature-weighting technique that yields consistently high accuracies, always approaching or exceeding those of the other algorithms we considered.

There is much room for future work. First, these results should be validated with other datasets. Second, we will investigate other relational features, such as those that are non-binary or based on links that are more complex (e.g., co-citations) or are inferred from other content. Third, we considered only one collective inference technique, ICA. Other robust techniques (e.g., Gibbs sampling) deserve examination, and ICA itself could also be refined.

Finally, we plan to examine how collective inference could be applied to problem-solving tasks other than classification (e.g., diagnosis, planning). Collective inference can be viewed as a process for incorporating background knowledge (represented explicitly or otherwise inferred) into a reasoning task. This relational knowledge, which is available in many applications, may also be useful for these other types of problem-solving tasks.

Acknowledgements

We thank the Naval Research Laboratory and the United States Naval Academy for supporting this research. Thanks to Derek Bridge, Antal van den Bosch, and the anonymous reviewers for their help in improving this paper. Thanks to Lise Getoor and Prithviraj Sen for providing the Cora and CiteSeer datasets. Portions of this analysis were conducted using Proximity, an open source software environment developed at the University of Massachusetts, Amherst.

References

Aha, D.W., and Marling, C. (Eds.) (2005). Special issue on Case-Based Reasoning. *Knowledge Engineering Review*, **20**(3).

Bergmann, R., Kolodner, J., and Plaza, E. (2005). Representation in case-based reasoning. *Knowledge Engineering Review*, **20**(3), 209-213.

Chakrabarti, S., Dom, B., and Indyk, P. (1998). Enhanced hypertext categorization using hyperlinks. *Proceedings of the International Conference on Management of Data* (pp. 307-318). Seattle, WA: ACM.

CiteSeer (2006). CiteSeer.IST scientific literature digital library. [http://citeseer.ist.psu.edu].

Craven, M., DiPasquo, D., Freitag, D, McCallum, A., Mitchell, T., Nigam, K., and Slattery, S. (1998). Learning to extract symbolic knowledge from the world wide web. *Proceedings of Fifteenth National Conference on Artificial Intelligence* (pp. 509-516). Madison, WI: AAAI.

Gupta K.M., Aha D.W., and Moore P.G. (2006). Rough set feature selection algorithms for textual case-based classification. *Proceedings of the Eighth European Conference on Case-based Reasoning* (pp. 166-181). Ölündeniz, Turkey: Springer.

Jensen, D. and Neville, J. (2002). Autocorrelation and linkage cause bias in evaluation of relational learners. *Proceedings of the Twlevth International Conference on Inductive Logic Programming* (pp. 101-116). Sydney, Australia: Springer.

Jensen, D., Neville, J., and Gallagher, B. (2004). Why collective inference improves relational classification. *Proceedings of the Tenth International Conference on Knowledge Discovery and Data Mining* (pp. 593-598). Seattle, WA: ACM.

Kohavi, R., and John, G. (1997). Wrappers for feature subset selection. *Artificial Intelligence*, **97**, 273-324.

Lu, Q. and Getoor, L. (2003). Link-based classification. *Proceedings of the Twentieth International Conference on Machine Learning* (pp. 496-503). Washington, DC: AAAI.

McCallum, A. Nigam, K., Rennie, J., and Seymore, K. (2000). Automating the construction of internet portals with machine learning. *Information Retrieval*, **3**, 127-163.

Neville, J. and Jensen, D. (2000). Iterative classification in relational data. In L. Getoor and D. Jensen (Eds.) *Learning Statistical Models from Relational Data: Papers from the AAAI Workshop* (Technical Report WS-00-06). Austin, TX: AAAI.

Neville, J., and Jensen, D. (2003). Collective classification with relational dependency networks. In S. Dzeroski, L. De Raedt, and S. Wrobel (Eds.) *Multi-Relational Data Mining: Papers from the ICKDDM Workshop.* Washington, DC: ACM.

Neville, J. and Jensen, D. (2005). Leveraging relational autocorrelation with latent group models. *Proceedings of the Fifth International Conference on Data Mining* (pp. 322-329). Houston, TX: IEEE.

Sen, P and Getoor, L. (2006). Empirical comparison of approximate inference algorithms for networked data. In A. Fern, L. Getoor, and B. Milch (Eds.) *Open Problems in Statistical Relational Learning: Papers from the ICML Workshop.* Pittsburgh, PA: www.cs.umd.edu/projects/srl2006.

Taskar, B., Abbeel, P., and Koller, D. (2002). Discriminative probabilistic models for relational data. *Proceedings of the Eighteenth Conference on Uncertainty in Artificial Intelligence* (pp. 485-492). Edmonton, Canada: Morgan Kaufmann.

Weber, R.O., Ashley, K.D., and Brüninghaus, S. (2005). Textual case-based reasoning. *Knowledge Engineering Review*, **20**(3), 255-260.

Wiratunga, N., Koychev, I., and Massie, S. (2004). Feature selection and generalization for retrieval of textual cases. *Proceedings of the Seventh European Conference on Case-Based Reasoning* (pp. 806-820). Madrid, Spain: Springer.

Yang, Y., Slattery, S., and Ghani, R. (2002). A study of approaches to hypertext categorization. *Journal of Intelligent Information Systems,* **18**(2-3), 219-241.