Detecting and Revising Misclassifications Using ILP

Masaki Yokoyama, Tohgoroh Matsui, and Hayato Ohwada

Department of Industrial Administration, Faculty of Science and Technology,
Tokyo University of Science,
2641 Yamazaki, Noda-shi, Chiba 278-8510, Japan
j7404659@ed.noda.tus.ac.jp
{matsui, ohwada}@ia.noda.tus.ac.jp

Abstract. This paper proposes a method for detecting misclassifications of a classification rule and then revising them. Given a rule and a set of examples, the method divides misclassifications by the rule into *miscovered* examples and *uncovered* examples, and then, separately, learns to detect them using Inductive Logic Programming (ILP). The method then combines the acquired rules with the initial rule and revises the labels of misclassified examples. The paper shows the effectiveness of the proposed method by theoretical analysis. In addition, it presents experimental results, using the Brill tagger for Part-Of-Speech (POS) tagging.

1 Introduction

Classification is one of the most popular fields in machine learning. It is concerned with constructing new classification rules from given training examples. Most previous work has focused on creating rules from scratch. Therefore, these approaches do not make use of previously constructed classification rules, even if they are reasonable. We consider that such rules are useful, and that it is more effective to correct misclassifications of a rule, than to create a new classification rule from scratch.

In this paper, we propose a method that detects misclassifications of a classification rule and then revises them. Given a rule and a set of examples, the method divides misclassifications by the rule into *miscovered* examples and *uncovered* examples and, separately, learns to detect them. It then combines the acquired rules with the initial rule and revises the labels of misclassified examples. This paper shows the effectiveness of the proposed method by theoretical analysis.

We use Inductive Logic Programming (ILP) to learn rules for detecting and revising misclassifications. ILP is a framework that combines machine learning and logic programming. ILP systems construct logic programs from examples and from background knowledge, which is also described by logic programs. One of the most important advantages of using ILP for discovering knowledge is that ILP can acquire hypotheses that can be understood by human beings. Another important advantage of ILP is that it is able to use background knowledge.

We have applied our method to Part-Of-Speech (POS) tagging, to which ILP has been applied previously [1]. We use the Brill tagger [2] as the initial classifier, which is one of the best rule-based tagging systems and is widely used in research into natural

A. Hoffmann, H. Motoda, and T. Scheffer (Eds.): DS 2005, LNAI 3735, pp. 363–370, 2005.

language processing. This paper shows the results of combining the Brill tagger with the additional acquired rules.

2 Miscovered Examples and Uncovered Examples

In this paper, we consider binary classification, which is also called concept learning. Let x be an example from a set of possible examples $\mathcal{X}$. The example is expressed as $(x, c(x))$, where c is a target function. If x belongs to the target concept, then $c(x) = 1$; if otherwise, $c(x) = 0$.

Misclassified examples of a classification rule are either *miscovered* examples or *uncovered* examples. Consider a classification rule r. Let h_r be the hypothesis function of r: if it estimates that x belongs to the target concept, then $h_\mathrm{r}(x) = 1$; otherwise, $h_\mathrm{r}(x) = 0$. We say that an example $x \in \mathcal{X}$ is *miscovered* by a classification rule r whenever $c(x) = 0$, but $h_\mathrm{r}(x) = 1$. We say that x is *uncovered* by r whenever $c(x) = 1$, but $h_\mathrm{r}(x) = 0$. Fig. 1 shows miscovered examples and uncovered examples of a classification rule r for a target concept. Miscovered examples and uncovered examples are sometimes called false positives and false negatives, respectively.

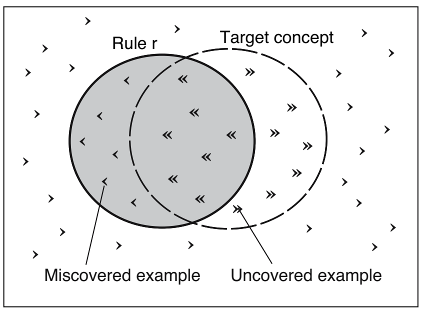

Fig. 1. Miscovered examples and Uncovered examples of a Classification Rule r for a Target concept

3 Method

3.1 Detecting and Revising Miscovered Examples

First, we consider the detection and revision of miscovered examples by using ILP. We generate examples for ILP from the data set by using the initial classification rule. We then construct a rule for detecting miscovered examples. Finally, we revise the labels of the detected miscovered examples.

Consider a classification rule r. Because all of the examples miscovered by r are included in examples covered by r, we can define the problem of detecting miscovered examples as follows: given a classification rule r and an example x that is covered by r, estimate whether x is miscovered or not.

Denote the subset of training examples that are covered by r as $\mathcal{E}_{\mathrm{m}}$. We then divide them into miscovered and correctly covered examples. Let $\mathcal{E}_{\mathrm{m}}^{+}$ be the set of miscovered examples, and let $\mathcal{E}_{\mathrm{m}}^{-}$ be the set of correctly covered examples. $\mathcal{E}_{\mathrm{m}}^{+}$ and $\mathcal{E}_{\mathrm{m}}^{-}$ can be written as:

$$\mathcal{E}_{\mathrm{m}}^{+} = \{x \mid (x, c(x)) \in \mathcal{D},\ h_{\mathrm{r}}(x) = 1,\ c(x) = 0\},$$
$$\mathcal{E}_{\mathrm{m}}^{-} = \{x \mid (x, c(x)) \in \mathcal{D},\ h_{\mathrm{r}}(x) = 1,\ c(x) = 1\},$$

where $\mathcal{D}$ is the set of training examples, h_{r} is the estimating function of r, and c is the target-concept function. This is shown in the left hand figure in Fig. 2, where the $+$ signs are positive examples and $-$ signs are negative examples.

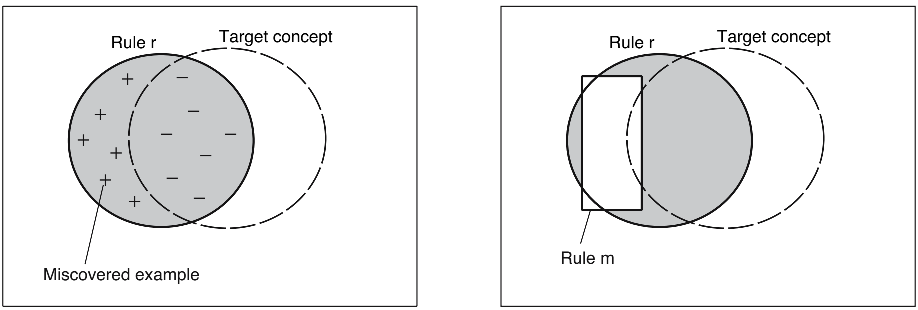

Fig. 2. Training examples for the miscovered concept (left) and the combined classification rule, h_{rm}, of the acquired rule and the initial rule (right)

Next, using ILP, we acquire a hypothesis $\mathcal{H}_{\mathrm{m}}$ from $\mathcal{E}_{m}^{+}$, $\mathcal{E}_{m}^{-}$, and background knowledge $\mathcal{B}$, such that $\mathcal{B} \vee \mathcal{H}_{\mathrm{m}} \models \mathcal{E}_{m}^{+}$ and $\mathcal{B} \vee \mathcal{H}_{\mathrm{m}} \not\models \mathcal{E}_{m}^{-}$. We define the estimating function h_{m} as: if $\mathcal{B} \vee \mathcal{H}_{\mathrm{m}} \models x$ for an example $x \in X$, then $h_{\mathrm{m}}(x) = 1$; otherwise, $h_{\mathrm{m}}(x) = 0$.

After acquiring $\mathcal{H}_{\mathrm{m}}$, we revise the misclassified labels by combining h_{r} with h_{m}. We define the combined hypothesis function h_{rm} as:

$$h_{\mathrm{rm}}(x) = \begin{cases} 1 & \text{if } h_{\mathrm{r}}(x) = 1 \text{ and } h_{\mathrm{m}}(x) = 0, \\ 0 & \text{otherwise.} \end{cases}$$

The right-hand figure of Fig. 2 illustrates this combined classification rule rm. If an example is included in the shaded area, the classification rule now estimates that it belongs to the target concept.

3.2 Detecting and Revising Uncovered Examples

We now consider uncovered examples. Again, we generate examples for detection and then revision. Previously, we used examples covered by r as a source of miscovered examples, but now we use the remaining examples, i.e., examples not covered by r. Denote the subset of training examples that are not covered by r as $\mathcal{E}_{\mathrm{u}}$. We divide these

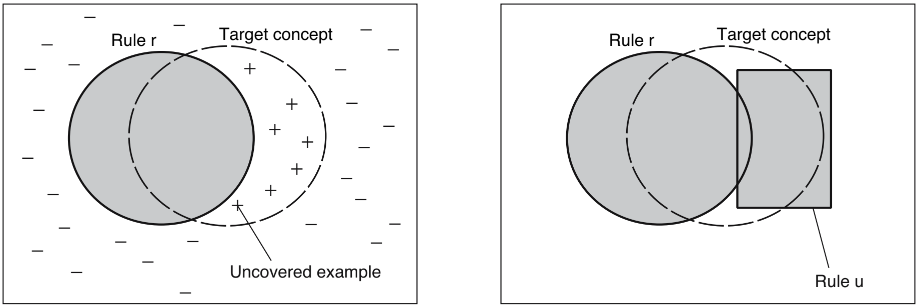

Fig. 3. Training examples for the uncovered concept (left) and the combined classification rule, h_{ru}, of the acquired rule and the initial rule (right)

examples into two subsets. Let $\mathcal{E}_{\mathrm{u}}^{+}$ be the set of uncovered examples, and let $\mathcal{E}_{\mathrm{u}}^{-}$ be the set of correctly not-covered examples. $\mathcal{E}_{\mathrm{u}}^{+}$ and $\mathcal{E}_{\mathrm{u}}^{-}$ can be written as:

$$\mathcal{E}_{\mathrm{u}}^{+} = \{x \mid (x, c(x)) \in \mathcal{D},\ h_{\mathrm{r}}(x) = 0,\ c(x) = 1\}\,,$$
$$\mathcal{E}_{\mathrm{u}}^{-} = \{x \mid (x, c(x)) \in \mathcal{D},\ h_{\mathrm{r}}(x) = 0,\ c(x) = 0\}\,.$$

The left-hand figure of Fig. 3 shows these training examples $\mathcal{E}_{\mathrm{u}}^{+}$ and $\mathcal{E}_{\mathrm{u}}^{-}$.

We now construct a hypothesis $\mathcal{H}_{\mathrm{u}}$ from $\mathcal{E}_{\mathrm{u}}^{+}$, $\mathcal{E}_{\mathrm{u}}^{-}$, and background knowledge $\mathcal{B}$, using ILP. We define the estimating function as h_{u}: $h_{\mathrm{u}}(x) = 1$ if $\mathcal{B} \vee \mathcal{H}_{\mathrm{u}} \models x$ for an example $x \in X$; otherwise, $h_{\mathrm{u}}(x) = 0$. After acquiring $\mathcal{H}_{\mathrm{u}}$, we revise the misclassified labels by combining h_{r} with h_{u}. We define the combined hypothesis function h_{ru} as:

$$h_{\mathrm{ru}}(x) = \begin{cases} 1 & \text{if } h_{\mathrm{r}}(x) = 1 \text{ or } h_{\mathrm{u}}(x) = 1, \\ 0 & \text{otherwise.} \end{cases}$$

The right-hand figure of Fig. 3 illustrates this classification rule ru.

3.3 Detecting and Revising Misclassified Examples

Finally, we combine the two acquired hypotheses with the initial classification rule. Because h_{m} and h_{u} are constructed from nonoverlapping training sets, we can combine them directly. We define a combined estimating function h_{rmu}:

$$h_{\mathrm{rmu}}(x) = \begin{cases} 1 & \text{if } h_{\mathrm{r}}(x) = 1 \text{ and } h_{\mathrm{m}}(x) = 0, \text{ or } h_{\mathrm{r}}(x) = 0 \text{ and } h_{\mathrm{u}}(x) = 1 \\ 0 & \text{otherwise.} \end{cases}$$

Fig. 4 illustrates this final combined classification rule h_{rmu}. Given an example x, we firstly compute $h_{\mathrm{r}}(x)$. If we find that $h_{\mathrm{r}}(x) = 1$, then we calculate $h_{\mathrm{m}}(x)$; otherwise, we calculate $h_{\mathrm{u}}(x)$. Thus, we choose the second classification rule depending on the situation, and it revises labels that were misclassified by the initial classification rule.

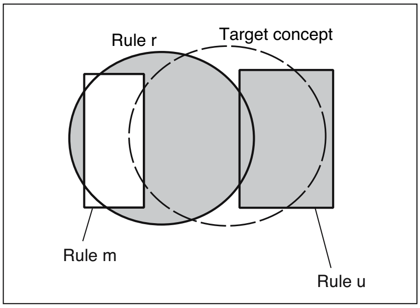

Fig. 4. The final combined classification rule h_{rmu}

4 Theoretical Analysis

We can show the effectiveness of the proposed method by theoretical analysis.

Theorem 1. *Let P_r and A_r be the precision and the accuracy of rule r. If the inequality $P_{\mathrm{m}} \geq 1/2$ is satisfied, then the inequality $A_{\mathrm{rm}} \geq A_{\mathrm{r}}$ is valid.*

Proof. To prove the theorem, consider the difference:

$$A_{\mathrm{rm}} - A_{\mathrm{r}} = \frac{|\mathrm{TP}_{\mathrm{rm}}| + |\mathrm{TN}_{\mathrm{rm}}|}{|\mathcal{E}_{\mathrm{rm}}|} - \frac{|\mathrm{TP}_{\mathrm{r}}| + |\mathrm{TN}_{\mathrm{r}}|}{|\mathcal{E}_{\mathrm{r}}|},$$

where $\mathcal{E}_{\mathrm{rm}}$ and $\mathcal{E}_{\mathrm{r}}$ are the example sets for rm and r, respectively. Since the example sets are the same, the denominators are the same, and positive. Now consider the numerators. In our method, examples classified by the rule rm can be written as:

$$\mathrm{TP}_{\mathrm{rm}} = \mathrm{TP}_{\mathrm{r}} \setminus \mathrm{FP}_{\mathrm{m}} \qquad \mathrm{FP}_{\mathrm{m}} \subseteq \mathrm{TP}_{\mathrm{r}}, \tag{1}$$

$$\mathrm{TN}_{\mathrm{rm}} = \mathrm{TN}_{\mathrm{r}} \cup \mathrm{TP}_{\mathrm{m}} \qquad \mathrm{TN}_{\mathrm{r}} \cap \mathrm{TP}_{\mathrm{m}} = \emptyset, \tag{2}$$

where TP_r, FP_r, FN_r, and TN_r are sets of true positive, false positive, false negative, and true negative examples of r, respectively. From Equations (1) and (2), the inequality

$$\begin{aligned}
&|\mathrm{TP}_{\mathrm{rm}}| + |\mathrm{TN}_{\mathrm{rm}}| - (|\mathrm{TP}_{\mathrm{r}}| + |\mathrm{TN}_{\mathrm{r}}|) \\
&= |\mathrm{TP}_{\mathrm{r}} \setminus \mathrm{FP}_{\mathrm{m}}| + |\mathrm{TN}_{\mathrm{r}} \cup \mathrm{TP}_{\mathrm{m}}| - (|\mathrm{TP}_{\mathrm{r}}| + |\mathrm{TN}_{\mathrm{r}}|) \\
&= (|\mathrm{TP}_{\mathrm{r}}| - |\mathrm{FP}_{\mathrm{m}}|) + (|\mathrm{TN}_{\mathrm{r}}| + |\mathrm{TP}_{\mathrm{m}}|) - (|\mathrm{TP}_{\mathrm{r}}| + |\mathrm{TN}_{\mathrm{r}}|) \\
&= |\mathrm{TP}_{\mathrm{m}}| - |\mathrm{FP}_{\mathrm{m}}| = |\mathrm{TP}_{\mathrm{m}}| \, \frac{2\,(P_{\mathrm{m}} - 1/2)}{P_{\mathrm{m}}} \geq 0
\end{aligned}$$

is valid, if the condition of the theorem is satisfied. The theorem is proved.

Theorem 2. *If the inequality $P_{\mathrm{u}} \geq 1/2$ is satisfied, then the inequality $A_{\mathrm{rmu}} \geq A_{\mathrm{rm}}$ is valid.*

This proof is omitted, to save space.

Finally, the following theorem indicates the effectiveness of our method:

Theorem 3. *If the inequalities $P_{\mathrm{m}} \geq 1/2$ and $P_{\mathrm{u}} \geq 1/2$ are satisfied, then the inequality $A_{\mathrm{rmu}} \geq A_{\mathrm{r}}$ is valid.*

Proof. From Theorems 1 and 2, $A_{\mathrm{rm}} \geq A_{\mathrm{r}}$ and $A_{\mathrm{rmu}} \geq A_{\mathrm{rm}}$ are valid, if the conditions of the theorem are satisfied. Therefore, the inequality $A_{\mathrm{rmu}} \geq A_{\mathrm{rm}} \geq A_{\mathrm{r}}$ is valid, if the conditions of the theorem are satisfied. The theorem is proved.

Since it is not difficult to learn a classifier whose precision is greater than or equal to $1/2$ in binary classification problems, the classification accuracy of our method can be higher than that of the initial classification rule.

5 Experiment: Part-of-Speech Tagging

5.1 Accuracy Comparison

POS tagging is the problem of assigning POS tags to each word in a document. We have applied our method to POS tagging, using the Brill tagger [2] as the initial classification rule. The data set is the set of Wall Street Journal articles in the Penn Treebank Project [3].

POS tagging involves more than three classes, and we adopted the one-against-the-rest method for formulation in terms of binary classification. Since there are 45 kinds of tags, we created 45 binary classification problems. For each problem, we applied the Brill tagger and created examples for learning the concepts of miscovered examples and uncovered examples. We used an ILP system, GKS [4,5], to learn the concepts with an acceptable error ratio of 0.2. We prepared the background knowledge of referring to the preceding three words and the following three words. We evaluated the performance of the acquired rules with 10-fold cross validation. We compared the accuracy of the initial classification rule of the Brill tagger with that of the proposed method. In this experiment, we added true-positive examples of the Brill tagger to the negative training examples for the uncovered concept. This enables us to acquire a hypothesis that covers only the uncovered examples. We also proved that Theorem 2 is true in this case.

Table 1 shows the results for each tag and overall. A_{r} stand for the accuracy of the Brill tagger alone. A_{rmu} stand for that of the combined classification rule, using the proposed method. P_{m} and P_{u} are the precisions of m and u alone, respectively. The "-" symbol means that the ILP system could not acquire rules at all. For all of the tags, the accuracies of the proposed method, A_{rmu}, were better than or equal to those of the Brill tagger alone, A_{r}. Because P_{m} and P_{u} were greater than $1/2$, the conditions of Theorem 3 were satisfied.

5.2 Discovered Knowledge on Misclassifications

There is another good aspect of the proposed method, in addition to increased accuracy: we have human-readable acquired knowledge on misclassifications, because ILP can create a hypothesis represented by first-order logic.

Here is the acquired knowledge for the "preposition" tag. The Prolog-formatted rule for the miscovered examples was as follows:

Table 1. The experiment result

Tag	A_{r}	A_{rmu}	P_{m}	P_{u}	Tag	A_{r}	A_{rmu}	P_{m}	P_{u}
cc	**0.9998**	**0.9998**	0.8889	-	pp	**0.9998**	**0.9999**	1.0	-
cd	**0.9991**	**0.9995**	1.0	0.9297	ppz	**0.9999**	**1.0**	-	1.0
cln	**0.9999**	**0.9999**	-	-	rb	**0.9947**	**0.9963**	0.9005	0.9488
cma	**0.9999**	**0.9999**	-	-	rbr	**0.9989**	**0.9992**	0.8682	0.9296
dlr	**1.0**	**1.0**	-	-	rbs	**0.9995**	**0.9999**	1.0	0.9482
dt	**0.9920**	**0.9988**	0.7778	0.9360	rp	**0.9984**	**0.9984**	-	-
ex	**0.9999**	**0.9999**	-	0.8472	rpn	**0.9988**	**0.9988**	-	-
fw	**0.9998**	**0.9999**	1.0	0.8710	rqt	**0.9999**	**0.9999**	0.8824	-
in	**0.9907**	**0.9943**	0.9947	0.9716	stp	**0.9999**	**0.9999**	-	-
jj	**0.9892**	**0.9924**	0.7888	0.9005	sym	**0.9987**	**0.9999**	-	0.9565
jjr	**0.9991**	**0.9993**	0.8788	0.8310	to	**0.9999**	**0.9999**	-	-
jjs	**0.9995**	**0.9996**	1.0	0.7640	uh	**0.9999**	**0.9999**	0.8000	-
lpn	**1.0**	**1.0**	-	-	vb	**0.9950**	**0.9974**	0.6429	0.8627
lqt	**1.0**	**1.0**	-	-	vbd	**0.9938**	**0.9949**	0.9162	0.9043
ls	**0.9999**	**0.9999**	-	-	vbg	**0.9976**	**0.9982**	0.6712	0.8708
md	**0.9999**	**0.9999**	-	-	vbn	**0.9924**	**0.9953**	0.7073	0.8614
nn	**0.9872**	**0.9914**	0.8165	0.9088	vbp	**0.9953**	**0.9965**	0.9888	0.9203
nns	**0.9967**	**0.9982**	0.8354	0.9133	vbz	**0.9971**	**0.9976**	0.9212	0.8766
np	**0.9941**	**0.9961**	0.7720	0.9401	wdt	**0.9976**	**0.9980**	0.9405	0.9730
nps	**0.9976**	**0.9978**	0.7024	0.8773	wp	**0.9999**	**0.9999**	-	-
pdt	**0.9998**	**0.9998**	0.8947	-	wpz	**1.0**	**1.0**	-	-
pnd	**1.0**	**1.0**	-	-	wrb	**0.9999**	**0.9999**	-	-
pos	**0.9986**	**0.9999**	-	0.9642	**All**	**0.9978**	**0.9986**	0.8973	0.9151

```
miscovered(A) :- post1word(A,'.').
miscovered(A) :- post2tag(A,vb), word(A,'like').
```

This rule means that the given word A is a miscovered example, i.e., it is not a preposition if: the following word is "." (period sign); or the next-but-one word is tagged "vb" and the given word is "like." Therefore, we can discover the Brill tagger mistakes with respect to prepositions. For example, the Brill tagger sometimes classifies the final word of a sentence as a preposition.

Similarly, we can see the rule for the uncovered examples. The rule is as follows:

```
uncovered(A) :- word(A,'up').
uncovered(A) :- post3word(A,'different').
```

This means that the given word A is an uncovered example, i.e. it is also a preposition if: the given word is "up", or the third-next word is "different".

We consider these rules to be very useful for correcting the Brill tagger itself. They show where we should change the Brill tagger's rule. So, if we install this knowledge into the Brill tagger, its performance will improve.

6 Conclusion

This paper proposes a method for decreasing misclassification, by using ILP to detect and revise misclassifications. The proposed method acquires two additional classification rules and combines them with the initial classification rule. We then show, by theoretical analysis, that this method works well. Finally, we apply it to POS tagging and present the experimental results.

Abney et al. have applied boosting to tagging [6]. They used their algorithm, AdaBoost, which calls a weak learner repeatedly to update the weights of examples. If the hypothesis acquired by the weak learner incorrectly classifies an example, it increases the weight; otherwise, it decreases the weight. Given an example to be predicted, boosting produces the final label, using a simple vote of the weak hypotheses. Although it can improve the classification accuracy very well, it cannot provide an understandable final hypothesis.

The good points of our method are that:

- it is simple and reliable,
- it can reduce the misclassification produced by the initial classification rule,
- it is shown that the classification accuracy of our method can be higher than that of initial classification rule, and
- the acquired rules are useful for modifying the initial rule because of their readability due to the use of ILP.

One drawback of our method is that it tends to overfit the training examples. Future work will include evaluating the acquired rules used to modify the initial classification rules.

References

1. James Cussens. Part-of-speech tagging using progol. S. Džeroski and N. Lavrač, editors, In *Proceedings of the 7th International Workshop on Inductive Logic Programming*, volume 1297, pages 93–108. Springer-Verlag, 1997.
2. Eric Brill. Some advances in transformation-based part of speech tagging. In *Proceedings of the 12th National Conference on Artificial Intelligence (AAAI-94)*, volume 1, pages 722–727, 1994.
3. Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. Building a large annotated corpus of english: the penn treebank. *Computational Linguistics*, 19(2):313–330, 1993.
4. Fumio Mizoguchi and Hayato Ohwada. Constrained relative least general generalization for inducing constraint logic programs. *New Generation Computing*, 13:335–368, 1995.
5. Fumio Mizoguchi and Hayato Ohwada. Using inductive logic programming for constraint acquisition in constraint-based problem solving. In *Proceedings of the 5th International Workshop on Inductive Logic Programming*, pages 297–322, 1995.
6. S. Abney, R. Schapire, and Y. Singer. Boosting applied to tagging and pp attachment. In *Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, 1999.