ILP を用いた分類器の誤分類の判別方法

A method for detecting misclassifications of classifier using ILP.

横山 正樹 † 松井 藤五郎 ‡ 大和田 勇人 ‡
Masaki YOKOYAMA† Tohgoroh MATSUI‡ Hayato OHWADA ‡

† 東京理科大学大学院 理工学研究科 経営工学専攻
Department of Industrial Administration, Graduate School of Science and Technology, Tokyo University of Science

‡ 東京理科大学 理工学部 経営工学科
Department of Industrial Administration, Faculty of Science and Technology, Tokyo University of Science
j7404659@ed.noda.tus.ac.jp {matsui, ohwada}@ia.noda.tus.ac.jp

現在、存在している分類器は誤分類を含んでしまい, 精度を 100%にすることは大変困難である. これに対して, 統計や機械学習の手法を用いて, 新たな分類器を構築し, 精度を向上させる動きが見られる. しかしながら, 既存の分類器は分野ごとの専門家を通じて分類ルールを定式化している. したがって, 新しい分類器を構築することは, たとえ分類器のルールが有用であったとしても, 無視してしまう. そこで, 本稿では, 既存の分類器の誤分類を判別し, 修正することにより精度を向上させる手法を提案する. そして, 我々は 2 つの実験を行い, 結果を考察することでどちらの方法がよいかを決定する. また, Inductive Logic Programming (ILP) を用いることにより, 既存の分類器がどのような誤分類を起こすかに関するルールを導く.

1 はじめに

近年, さまざまな分野において分類問題が行われている. 分類問題とは事物の性質に基づいて事例を分類する問題である. この分類問題に関して, 自動的に分類できる分類器が分野ごとに存在する. 現在存在している分類器は大変精度が高いものが多い. しかしながら, 分類器は誤分類を含んでしまい, 精度を 100%にすることは大変困難である. これに対して, 統計や機械学習の手法を用いて, 新たな分類器を構築し, 精度を向上させる動きが見られる. しかしながら, 既存の分類器は分野ごとの専門家を通じて分類ルールを定式化している. したがって, 新しい分類器を構築することは, たとえ分類器のルールが有用であったとしても, 無視してしまう. そこで, 本稿では, 既存の分類器の誤分類を判別し, 修正することにより精度を向上させる手法を提案する. ここで, 我々は本手法の有効性を示すために 2 つの実験を行う. そして, 2 つの実験結果を考察することでどちらの方法がよいかを決定する. また, Inductive Logic Programming (ILP) を用いることにより, 既存の分類器がどのような誤分類を起こすかに関するルールを導く.

2 誤分類とは

本節では, 2 クラス分類問題に焦点を当てる. 2 クラス分類問題の場合, 一方のクラスに着目すれば, 「そのクラスに含まれる概念」を学習する問題として考えることができる. 事例 x の目標概念関数を c, 分類ルール r による分類評価関数を h_{r} とする. このとき, 分類結果は以下の 4 種類になる.

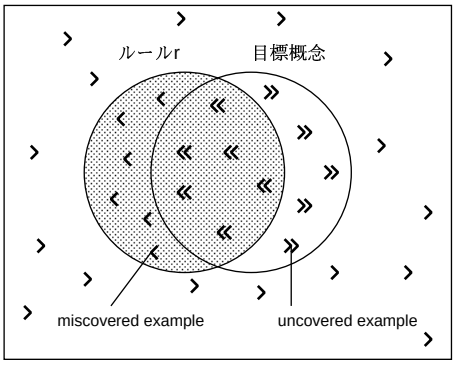

図 1: Miscovered example と uncovered example

1. $h_{\mathrm{r}}(x) = 1$ であり $c(x) = 1$ であるとき
2. $h_{\mathrm{r}}(x) = 1$ であるが $c(x) = 0$ であるとき
3. $h_{\mathrm{r}}(x) = 0$ であるが $c(x) = 1$ であるとき
4. $h_{\mathrm{r}}(x) = 0$ であり $c(x) = 0$ であるとき

ここで, 1 と 4 は正しく分類されている. しかしながら, 2 と 3 は誤って分類している. このとき, 2 では, 本来 h_{r} が x をカバーしてはならないが, カバーしてしまったところから事例 x を $miscovered\ example$

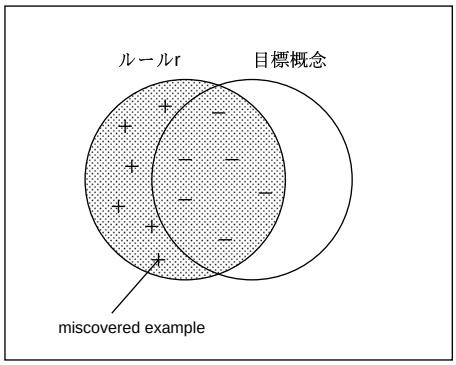

図 2: Miscovered examples を判別するための事例セット. “+” は正事例, “-” は負事例とおく

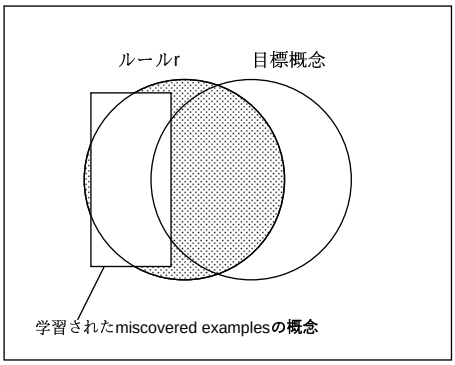

図 3: Miscovered examples の概念

と名付ける. また, 3 では, h_{r} が x をカバーしなければならないが, カバーしなかったところから事例 x を $uncovered\ example$ と名付ける. Miscovered example と uncovered example は図 1 に示す. 1-4 はそれぞれ true positive ($\mathrm{TP_r}$), false positive ($\mathrm{FP_r}$), false negative ($\mathrm{FN_r}$), true negative ($\mathrm{TN_r}$) と呼ばれている.

3 誤分類の判別と修正

本節では, 誤分類の判別と修正の仕方を記述する.

3.1 Miscovered examples の判別と修正

ここでは, miscovered examples の判別と修正の方法について説明する.

まず, 分類ルール r に被覆された事例を全部取り出し, miscovered examples の判別ルールを学習するためのトレーニングセット $\mathcal{E}_{\mathrm{m}}$ とする. $\mathcal{E}_{\mathrm{m}}$ の中で miscovered examples の事例集合と正しく分類された事例集合に分割し, 図 2 のようにそれぞれ $\mathcal{E}^+_{\mathrm{m}}, \mathcal{E}^-_{\mathrm{m}}$ とする. つまり,

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

とする. ここで, $\mathcal{D}$ はトレーニングセットを表す.

次に, ILP を用いて miscovered examples の判別ルールを学習する. $\mathcal{E}^+_m$ を正事例, $\mathcal{E}^-_m$ を負事例とおき, 背景知識を $\mathcal{B}$ を使用し,

$$\mathcal{B} \vee \mathcal{H}_{\mathrm{m}} \models e\ for\ each\ e \in \mathcal{E}^+_{\mathrm{m}}$$
$$\mathcal{B} \vee \mathcal{H}_{\mathrm{m}} \not\models e\ for\ cach\ e \in \mathcal{E}^-_{\mathrm{m}}$$

となるような仮説 $\mathcal{H}_{\mathrm{m}}$ を学習する. 図 3 は学習した概念を表す. ここで, 事例 $x \in X$ に関して $\mathcal{B} \vee \mathcal{H}_{\mathrm{m}} \models \mathcal{E}^+_{\mathrm{m}}$ となるとき, $h_{\mathrm{m}}(x) = 1$ となり, その他の場合, $h_{\mathrm{m}}(x) = 0$ となるような評価関数 h_{m} を定義する. $h_{\mathrm{m}}(x) = 1$ のとき, x は miscovered example と判別されたため, $h_{\mathrm{r}}(x) = 1$ かつ $h_{\mathrm{m}(x)} = 1$ のとき, 0 に再分類することで修正できる. つまり, miscovered examples を修正する評価関数は

$$h_{\mathrm{rm}}(x) = \begin{cases} 1 & \text{if } h_{\mathrm{r}}(x) = 1 \text{ かつ } h_{\mathrm{m}}(x) = 0, \\ 0 & \text{上記以外} \end{cases}$$

となり, 分類結果は

$$\mathrm{TP_{rm}} = \mathrm{TP_r} \setminus \mathrm{FP_m}, \quad \mathrm{FP_{rm}} = \mathrm{FP_r} \setminus \mathrm{TP_m},$$
$$\mathrm{FN_{rm}} = \mathrm{FN_r} \cup \mathrm{FP_m}, \quad \mathrm{TN_{rm}} = \mathrm{TN_r} \cup \mathrm{TP_m}.$$

となる.

3.2 Uncovered examples の判別と修正

次に, uncovered examples の判別と修正の方法について記述する.

まず, 分類ルール r に被覆されなかった事例を全部取り出し, uncovered examples の判別ルールを学習するためのトレーニングセット $\mathcal{E}_{\mathrm{u}}$ とする. そして, $\mathcal{E}_{\mathrm{u}}$ の中で, uncovered examples の事例集合と正しく分類された事例集合に分割し, 図 4 のようにそれぞれ $\mathcal{E}^+_{\mathrm{u}}$, $\mathcal{E}^-_{\mathrm{u}}$ とする. つまり,

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

とする.

次に, $\mathcal{E}^+_u$ を正事例, $\mathcal{E}^-_u$ を負事例とおき, 背景知識 $\mathcal{B}$ を使用し,

$$\mathcal{B} \vee \mathcal{H}_{\mathrm{u}} \models e\ for\ each\ e \in \mathcal{E}^+_{\mathrm{u}}$$
$$\mathcal{B} \vee \mathcal{H}_{\mathrm{u}} \not\models e\ for\ cach\ e \in \mathcal{E}^-_{\mathrm{u}}$$

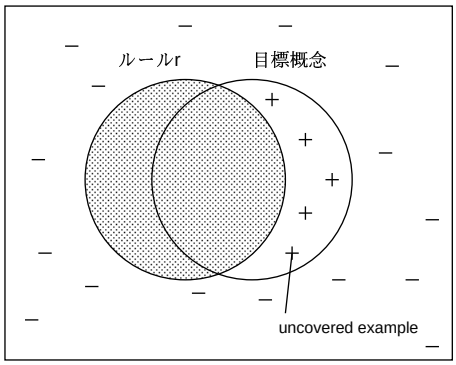

図 4: Uncovered examples を判別するための事例セット. “+” は正事例, “-” は負事例とおく

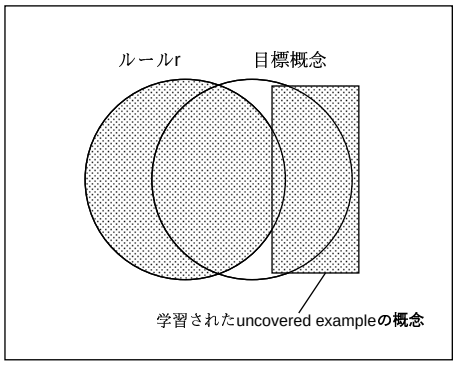

図 5: Uncovered examples の概念

となるような仮説 $\mathcal{H}_{\mathrm{u}}$ を学習する. 図 5 は学習した概念を示す. ここで, 事例 $x \in X$ に関して $\mathcal{B} \vee \mathcal{H}_{\mathrm{u}} \models \mathcal{E}_{\mathrm{u}}^{+}$ のとき, $h_{\mathrm{u}}(x) = 1$ となり, その他の場合, $h_{\mathrm{u}}(x) = 0$ となるような評価関数 h_{u} を定義する. このとき,

$$h_{\mathrm{ru}}(x) = \begin{cases} 1 & \text{if } h_{\mathrm{r}}(x) = 1 \text{ または } h_{\mathrm{u}}(x) = 1, \\ 0 & \text{上記以外} \end{cases}$$

となり, 分類結果は,

$$\mathrm{TP}_{\mathrm{ru}} = \mathrm{TP}_{\mathrm{r}} \cup \mathrm{TP}_{\mathrm{u}}, \qquad \mathrm{FP}_{\mathrm{ru}} = \mathrm{FP}_{\mathrm{r}} \cup \mathrm{FP}_{\mathrm{u}},$$
$$\mathrm{FN}_{\mathrm{ru}} = \mathrm{FN}_{\mathrm{r}} \setminus \mathrm{TP}_{\mathrm{u}}, \qquad \mathrm{TN}_{\mathrm{ru}} = \mathrm{TN}_{\mathrm{r}} \setminus \mathrm{FP}_{\mathrm{u}}.$$

となる.

3.3 誤分類の修正

最後に, 図 6 のように誤分類を修正するために, 既存の分類ルールに miscovered examples の判別ルール h_{rm} と uncovered examples の判別ルール h_{ru} を統合する. また, 誤分類を修正した本手法の分類評価関数

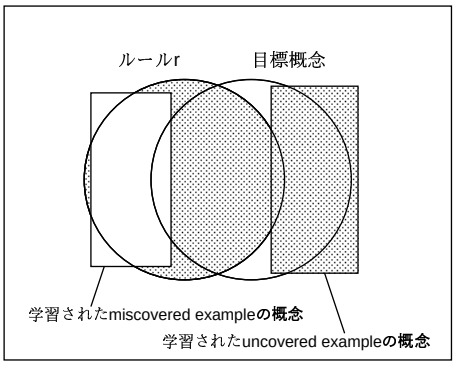

図 6: ルール r に miscovered examples と uncovered examples の概念を組み合わせたもの

h_{rmu} を以下のように定義する.

$$h_{\mathrm{rmu}}(x) = \begin{cases} 1 & \text{if } h_{\mathrm{r}}(x) = 1 \text{ かつ } h_{\mathrm{m}}(x) = 0, \text{ または} \\ & h_{\mathrm{r}}(x) = 0 \text{ かつ } h_{\mathrm{u}}(x) = 1 \\ 0 & \text{上記以外} \end{cases}$$

分類結果は

$$\mathrm{TP}_{\mathrm{rmu}} = (\mathrm{TP}_{\mathrm{r}} \setminus \mathrm{FP}_{\mathrm{m}}) \cup \mathrm{TP}_{\mathrm{u}},$$
$$\mathrm{FP}_{\mathrm{rmu}} = (\mathrm{FP}_{\mathrm{r}} \setminus \mathrm{TP}_{\mathrm{m}}) \cup \mathrm{FP}_{\mathrm{u}},$$
$$\mathrm{FN}_{\mathrm{rmu}} = (\mathrm{FN}_{\mathrm{r}} \cup \mathrm{FP}_{\mathrm{m}}) \setminus \mathrm{TP}_{\mathrm{u}},$$
$$\mathrm{TN}_{\mathrm{rmu}} = (\mathrm{TN}_{\mathrm{r}} \cup \mathrm{TP}_{\mathrm{m}}) \setminus \mathrm{FP}_{\mathrm{u}}.$$

となる.

4 本手法から導かれる定理

我々は, 本手法が精度を向上させるための条件を理論的に証明した. P_r, A_r をそれぞれ r の precision, accuracy とする.

定理 1 $P_{\mathrm{m}} \geq 1/2 \Leftrightarrow A_{\mathrm{rm}} \geq A_{\mathrm{r}}$

証明：rm, r における全事例を $\mathcal{E}_{\mathrm{rm}}, \mathcal{E}_{\mathrm{r}}$ とすると,

$$A_{\mathrm{rm}} \geq A_{\mathrm{r}}$$
$$\frac{|\mathrm{TP}_{\mathrm{rm}}| + |\mathrm{TN}_{\mathrm{rm}}|}{|\mathcal{E}_{\mathrm{rm}}|} \geq \frac{|\mathrm{TP}_{\mathrm{r}}| + |\mathrm{TN}_{\mathrm{r}}|}{|\mathcal{E}_{\mathrm{r}}|}$$
$$|\mathrm{TP}_{\mathrm{rm}}| + |\mathrm{TN}_{\mathrm{rm}}| \geq |\mathrm{TP}_{\mathrm{r}}| + |\mathrm{TN}_{\mathrm{r}}|$$
$$|\mathrm{TP}_{\mathrm{m}}| \geq |\mathrm{FP}_{\mathrm{m}}|$$
$$\frac{|\mathrm{TP}_{\mathrm{m}}|}{|\mathrm{FP}_{\mathrm{m}}|} + 1 \geq 2$$
$$1/2 \geq 1 - P_{\mathrm{m}}$$
$$P_{\mathrm{m}} \geq 1/2.$$

定理 2 $P_{\mathrm{u}} \geq 1/2 \Leftrightarrow A_{\mathrm{rmu}} \geq A_{\mathrm{rm}}$

証明は省略する.

定理 3 $P_{\mathrm{m}} \geq 1/2 \ \wedge \ P_{\mathrm{u}} \geq 1/2 \Leftrightarrow A_{\mathrm{rmu}} \geq A_{\mathrm{r}}$

証明：定理 1 と定理 2 より $P_{\mathrm{m}} \geq 1/2$ かつ $P_{\mathrm{u}} \geq 1/2$ が同時に満たされる場合, $A_{\mathrm{rmu}} \geq A_{\mathrm{rm}} \geq A_{\mathrm{r}}$ が成立する. また, その逆も成り立つ.

5 実験

ここでは, 本手法の有効性を示すために品詞 (POS) タグ付け問題に応用した. しかしながら, 品詞タグ付け問題は多クラス分類問題であるがゆえに, そのままでは本手法には応用できない.

そこで, 我々は *one against the rest* 法を用いた. この方法は, 一つのクラスと他の全部のクラスをもう一つのクラスとすることで 2 クラス化する. 例えば, その単語が「動詞である」クラスと「動詞ではない」クラスの 2 クラスと考える. そして, One against the rest 法を全クラスに行った. また, 本実験ではデータセットとして Penn TreeBank Project により作成されたタグ付きコーパスの Wall Street Journal (WSJ) を使用した [4]. WSJ は約百万単語で構成されており, 45 種類の品詞・記号に分類されている. 既存の分類器は, 品詞付与システムである Brill's Tagger [2] を用いた. Brill's Tagger はルールベースの分類器であり, 高い精度を示している. また, 誤分類の判別ルールを学習するために ILP システムは GKS [5, 6] を使用した. ratio パラメータは 0.2 とした. ILP における背景知識はその単語と前後 3 つまでの単語と品詞を用いた.

5.1 実験 1

Miscovered examples の判別では, まず miscovered example を全て取り出した. また, 時間短縮のため, miscovered examples と同じ数だけ $\mathrm{TP_r}$ に含まれる事例をランダムにサンプリングした. 次に, Miscovered examples を正事例, $\mathrm{TP_r}$ に含まれる事例を負事例とおき, ILP で学習した. そして学習されたルールに基づいて誤分類を判別し, 修正した. 同様に uncovered examples の判別においても, uncovered example を全部取り出し正事例, それと同じ数だけ $\mathrm{TN_r}$ の事例をランダムにサンプリングし負事例と設定し, ILP で学習した. ここで得られたルールに基づいて uncovered examples を判別し, 修正した. 評価は 10-fold cross validation で Brill's Tagger と Brill's Tagger の誤分類を修正した本手法の accuracy を比較した. 表 1 で, 実験結果を載せる. ここでは, タグごとの結果と全てのタグの合計した結果を載せる. また miscovered example と uncovered example の判別ルールがどれだけ正しく判別できていたかを表す precision を示した. 表 1 より, 全てのタグにおいて accuracy が下がることはなかった. それは $P_{\mathrm{m}} \geq 1/2$ かつ $P_{\mathrm{u}} \geq 1/2$ となり, Therem 3 の条件を満たすことができたからである. また, 正しく修正できた miscovered examples は 4,751 個で, 誤って miscovered examples と再分類してしまった事例は 544 個であった. Uncovered examles では, 45,221 個修正でき, 7,525 個誤って再分類してしまった.

5.2 実験 2

実験 1 の場合, uncovered examples の判別ルールは $\mathrm{TP_r}$ の事例を含んでしまう. そこで, 実験 2 では実験 1 の uncovered examples の判別において, $\mathrm{TP_r}$ の事例を負事例に追加した. これにより, uncovered examples だけを被覆するようなルールを学習することができる. 評価は実験 1 と同様に, 10-fold cross validation で Brill's Tagger の accuracy と本手法の accuracy を比較した. 表 2 で, 実験結果を掲載する. 表 2 から, 実験 2 においても, 全てのタグで本手法の精度は Brill's Tagger より下がることはなかった. つまり, 定理 3 の条件を満たすことができた. 実験 2 では, 36,415 個の uncovered examples を修正し, 3,379 個誤って再分類してしまった.

実験 1 の P_{u} は実験 2 の P_{u} よりも低いにもかかわらず, accuracy は実験 1 のほうが高い数値を示した. その理由は, accuracy で評価すると, 正しく修正できた個数が評価となる. つまり, 正しく分類できた事例の個数から誤って再分類してしまった事例の個数を引いた値が大きいほうが accuracy を向上させることができる. 実験 1 は $45,221 - 7,525 = 37,696$ となり, 実験 2 は $36,415 - 3,379 = 33,036$ となった. よって, 実験 1 のほうが accuracy が向上した. しかしながら, precision で評価すると, 正しく分類された割合で評価される. つまり, 誤って再分類した事例が多いと下がってしまう. 実験 1 は正しく分類できた事例は多いが, 誤って再分類してしまった事例も多い. したがって, 実験 2 に比べて P_{u} が低くなった.

ここで, 我々は uncovered examples だけを被覆するようなルールは Brill's Tagger の誤分類を起こすルー

表 1: 実験 1 の結果. A_{r}, A_{rmu} は Brill's Tagger と本手法の accuracy を表す. P_{m}, P_{u} は miscovered examples と uncovered examples の precision を示す. "-" は誤分類が少なくて ILP でルールが出なかったことを意味する.

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	0.8043	ppz	**0.9999**	**1.0**	-	0.8725
cln	**0.9999**	**0.9999**	-	-	rb	**0.9947**	**0.9964**	0.9005	0.8795
cma	**0.9999**	**0.9999**	-	-	rbr	**0.9989**	**0.9994**	0.8682	0.9145
dlr	**1.0**	**1.0**	-	-	rbs	**0.9995**	**1.0**	1.0	0.9912
dt	**0.9920**	**0.9992**	0.7778	0.9473	rp	**0.9984**	**0.9984**	-	-
ex	**0.9999**	**1.0**	-	1.0	rpn	**0.9988**	**0.9988**	-	-
fw	**0.9998**	**0.9999**	1.0	0.7795	rqt	**0.9999**	**0.9999**	0.8824	-
in	**0.9907**	**0.9943**	0.9947	0.8760	stp	**0.9999**	**0.9999**	-	-
jj	**0.9892**	**0.9937**	0.7888	0.8470	sym	**0.9987**	**0.9999**	-	0.9522
jjr	**0.9991**	**0.9996**	0.8788	0.8552	to	**0.9999**	**0.9999**	-	-
jjs	**0.9995**	**0.9996**	1.0	0.6598	uh	**0.9999**	**0.9999**	0.8000	-
lpn	**1.0**	**1.0**	-	-	vb	**0.9950**	**0.9977**	0.6429	0.8241
lqt	**1.0**	**1.0**	-	-	vbd	**0.9938**	**0.9950**	0.9162	0.8178
ls	**0.9999**	**0.9999**	-	-	vbg	**0.9976**	**0.9983**	0.6712	0.7726
md	**0.9999**	**0.9999**	-	-	vbn	**0.9924**	**0.9964**	0.7073	0.8078
nn	**0.9872**	**0.9916**	0.8165	0.8393	vbp	**0.9953**	**0.9966**	0.9888	0.8398
nns	**0.9967**	**0.9982**	0.8354	0.8328	vbz	**0.9971**	**0.9976**	0.9212	0.8056
np	**0.9941**	**0.9962**	0.7720	0.7783	wdt	**0.9976**	**0.9980**	0.9405	0.9474
nps	**0.9976**	**0.9980**	0.7024	0.7944	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**	-	0.9993	**All**	**0.9978**	**0.9987**	0.8973	0.8573

ルと言うこともできるため, 大変有用な知識を発見できると言える. そこで, 実験 2 のほうが優れているとする.

5.3 本手法から獲得できたルール

本手法で ILP を用いた利点は, ILP はルールを Prolog 形式でルールが導出されることである. これらは人間に理解可能である. ここで, 実験 2 で前置詞について獲得したルールを一部を以下に掲載する.

miscovered examples 判別ルール

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

これらは「次の単語が "." であるときその単語は miscovered example である」,「2 つ後ろの単語の品詞が "vb(動詞原形)" であり, 現在の単語が "like" であるときその単語は miscovered example である」ことを表す.

uncovered example 判別ルール

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

上記のルールは「その単語が "up" であるときその単語は uncovered example である」,「3 つ後ろの単語が "different" であるときその単語は uncovered example である」ことを意味する. これらは Brill's Tagger に組み込むことが可能であり, Brill's Tagger の精度を向上させるのに有効な知識であると考えられる.

6 関連研究

Karwath らの研究は分子生物学におけるタンパク質の相同性を推定することである [3]. これは相同性検索ツールの PSI-BLAST を使用し, 明らかに相同であると判断されたシーケンスを正事例とし, 相同でないと判断されたシーケンスを負事例として ILP で学習する. 学習された仮説を曖昧なシーケンスに適用することにより, PSI-BLAST よりも精度を向上させる手法を提案した. 実際に Recall は約 0.6%向上した.

また, Abney らは Boosting を品詞タグ付け問題に適用した [1]. Boosting は誤って分類した事例の重み

表 2: 実験 2 の結果. A_{r}, A_{rmu} は Brill's Tagger と本手法の accuracy を表す. P_{m}, P_{u} は miscovered examples と uncovered examples の precision を示す. "-" は誤分類が少なくて ILP でルールが出なかったことを意味する.

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

を高くし, 正しく分類した事例の重みを低くし, 再度学習器を構築する. この作業をくり返し, 投票を行い最終的なクラスを決定する. Abney らの手法の精度は大変高いものであった. しかしながら, Boosting は何回も学習し重みつきの投票を行うので複雑である. それに対して我々は, 誤分類の判別ルールに被覆された事例のみ修正するのでシンプルで分かり易いと言えるだろう.

7 まとめと今後の課題

本稿では, ILP を用いて既存の分類器の誤分類を判別し, 修正することにより精度を向上させる手法を提案した. 2 つの実験を行い, それぞれについて考察し, 実験 2 のほうがよいことを決定した. また, ILP を用いることにより, 既存の分類器がどのような誤分類を起こすかに関するルールを導くことができた.

我々は, 獲得できた誤分類の判別ルールを実際に Brill's Tagger に組み込むことが今後の課題である. 例えば,「次の単語が "." であるとき前置詞と分類する」のような新しい分類ルールを Brill's Tagger に追加することにより Brill's Tagger の精度を向上させることを考えている.

参考文献

[1] 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.

[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] Andreas Karwath and Ross D. King. Homology induction: the use of machine learning to improve sequence similarity searches. *BMC Bioinformatics*, 3(11), 2002.

[4] 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.

[5] Fumio Mizoguchi and Hayato Ohwada. Constrained relative least general generalization for inducing constraint logic programs. *New Generation Computing*, 13:335–368, 1995.

[6] 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.