社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2004－NL－161 (15)
2004／5／14

優先度学習を用いた自然言語処理

磯崎 秀樹　賀沢 秀人　平尾 努
日本電信電話株式会社 NTT コミュニケーション科学基礎研究所
〒 691-0237 京都府相楽郡精華町光台２－４
isozaki@cslab.kecl.ntt.co.jp

あらまし　最近、自然言語処理において SVM が利用されることが増えてきている。SVM は二値分類器としては高性能であるが、タスクによっては必ずして最良の方法とは限らない。たとえば英語依存構造解析は、各単語が同一文中のどの単語に係るかを決定するタスクであり、係り先として最良のものを選ぶ必要がある。またゼロ代名詞解消は、ゼロ代名詞が指している先行詞を推定するタスクであり、複数の指示対象の中から最良のものを選ぶ。これらのタスクは候補の優先度の比較が必要であり、SVM が最適とは言えない。そこで、この２つのタスクで SVM と優先度学習を用いた実験したところ、優先度学習の方が高精度であった。
キーワード　SVM、優先度学習、依存構造解析、ゼロ代名詞解消

Natural Language Processing based on Preference Learning

Hideki Isozaki　Hideto Kazawa　Tsutomu Hirao
NTT Communication Science Laboratories, NTT Corporation
2-4 Hikari-dai, Seika-cho, Soraku-gun, Kyoto, 619-0237

Abstract　Nowadays, SVM is widely used in Natural Language Processing. However, SVM is not necessarily the best learning method for some tasks. For instance, English word dependency analysis selects the best modificand for each word in a sentence. Zero pronoun resolution selects the best antecedent for each zero pronoun. SVM has shown good performance for these tasks. However, SVM is not designed for preference. Therefore, SVM is not optimal for these tasks. In this report, we show that Preference Learning gives better results than SVM for these tasks.
Keywords: SVM, preference learning, dependency analysis, zero pronoun resolution

1　はじめに

最近、依存構造解析やゼロ代名詞解析などのタスクにおいても SVM が利用されるようになってきている。これらは、複数の候補から最良のものを選び出すタスクである。依存構造解析では、係り先として最良のものが選ばれるのに対して、ゼロ代名詞解消では、先行詞の中で最良のものが選ばれる。

しかし、SVM は二値分類器として設計されたものであり、複数の候補の中から最良のものを選ぶためのものではない。一番簡単な方法は、SVM の出力する値の大小を比較して、最大の候補を選ぶというものである。

磯崎ら [7] はゼロ代名詞の解消のために、経験則で候補を並べておき、SVM の出力が最初に正になった候補を選んでいる。飯田ら [6] は２つの候補のどちらがいいかを選ぶことを繰り返すトーナメントモデルという方法を用いている。

工藤ら [11] は、日本語の性質を利用し、直後の文節に係るか否かの判断を繰り返すことで、日本語の依存構造木を作り出す。しかし、英語では前に係ることもあるので、山田ら [14] は、直前の単語に係るか否かと直後の単語に係るか否かの判断を繰り返す

ことで、英語の依存構造木を作り出す。

本稿では、英語依存構造解析とゼロ代名詞解消を対象として、複数の候補の中から最良のものを選ぶ、という基本に戻って、SVMのかわりに優先度学習(PL: Preference Learning)を採用することによって、SVMを超える成績が達成できることを示す。

2 SVMと優先度学習

ここでいう優先度学習 (Preference Learning) とは、SVMに簡単な変更を加えたものであり、Herbrich [4] によって提案され、Joachims [9] によって拡張された枠組みである。

SVMのトレーニングデータは $(y_i, \boldsymbol{x}_i)$ というペアで表される。ここで $\boldsymbol{x}_i$ はベクトルであり、$\boldsymbol{x}_i$ が正例の場合は $y_i = +1$、負例の場合は $y_i = -1$ となる。SVMは、以下の関数によって正例か負例かを判定する。

$$f(\boldsymbol{x}) = \sum_i^{\ell} y_i \alpha_i K(\boldsymbol{x}, \boldsymbol{x}_i) + b,$$

ここで $\{\alpha_i\}$ と b は定数、ℓ はトレーニングデータの数である。

SVMによる学習は以下の最大化に対応する [3]。

$$W(\alpha) = \sum_{i=1}^{\ell} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{\ell} \alpha_i \alpha_j y_i y_j K(\boldsymbol{x}_i, \boldsymbol{x}_j),$$

ここで $0 \leq \alpha_i \leq C$ かつ $\sum_{i=1}^{\ell} \alpha_i y_i = 0$。. なお、$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \phi(\boldsymbol{x}_i) \cdot \phi(\boldsymbol{x}_j)$ はあらかじめ定義されたカーネル関数である。$\phi(\boldsymbol{x})$ は $\boldsymbol{x}$ を高次元空間に写像する関数である。

これに対して、優先度学習のトレーニングデータは、$(y_i, \boldsymbol{x}_{i.1}, \boldsymbol{x}_{i.2})$ で表される。ここで $\boldsymbol{x}_{i.1}$ と $\boldsymbol{x}_{i.2}$ はベクトルであり、ペア $(\boldsymbol{x}_{i.1}, \boldsymbol{x}_{i.2})$ を表すのに $\boldsymbol{x}_{i.*}$ を用いる。$\boldsymbol{x}_{i.1}$ が $\boldsymbol{x}_{i.2}$ より良いとき $y_i = +1$ で、逆のときを $y_i = -1$ で表す。優先度学習では、SVMの $\phi(\boldsymbol{x})$ のかわりに、高次元空間におけるベクトルの差 $\phi(\boldsymbol{x}_{i.1}) - \phi(\boldsymbol{x}_{i.2})$ を用いる。

図1に1次元の場合を示す。この図では、「左=worse」「右=better」という優先度が成り立っているが、相対的なものなので、betterとworseの間に境界線 θ を引くことはできない。したがって、このようなタスクは、SVMが苦手とするタスクであると考えられる。しかし、betterのベクトルとworseのベクトルの差を学習する優先度学習では、簡単に扱える。

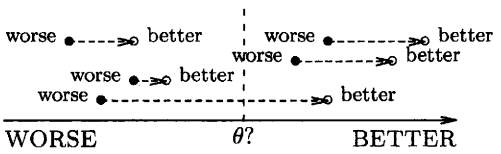

図 1: 一次元の優先度学習

ベクトル $\boldsymbol{x}$ の優先度は以下の式で与えられる。

$$g(\boldsymbol{x}) = \sum_i^{\ell} y_i \alpha_i (\phi(\boldsymbol{x}_{i.1}) - \phi(\boldsymbol{x}_{i.2})) \cdot \phi(\boldsymbol{x}).$$

$g(\boldsymbol{x}) > g(\boldsymbol{x}')$ が成り立てば、$\boldsymbol{x}$ は $\boldsymbol{x}'$ よりも優先される。優先度学習は以下の最大化に対応する。

$$W(\alpha) = \sum_{i=1}^{\ell} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{\ell} \alpha_i \alpha_j y_i y_j \mathcal{K}(\boldsymbol{x}_{i.*}, \boldsymbol{x}_{j.*})$$

ここで $0 \leq \alpha_i \leq C$ であり、また $\mathcal{K}(\boldsymbol{x}_{i.*}, \boldsymbol{x}_{j.*}) = (\phi(\boldsymbol{x}_{i.1}) - \phi(\boldsymbol{x}_{i.2})) \cdot (\phi(\boldsymbol{x}_{j.1}) - \phi(\boldsymbol{x}_{j.2}))$ である。なお、SVMでは $\sum_{i=1}^{\ell} \alpha_i y_i = 0$ という制約があった。しかし、この制約は優先度学習には適用されない。というのは、この制約は b から導かれたもので、優先度学習には b がないためである。したがって、SVMと違い、各 α_i は独立に変更できる。SVM$^{\text{light}}$ [8] には、一種の優先度学習が実装されているが、カーネルが使えない。

HerbrichやJoachimsは、情報検索における文書のランキングを想定している。とくにHerbrichのSupport Vector Ordinal Regression [5] は、5段階評価のようなものを想定している。これに対して、我々のアプリケーションでは、正解と不正解の2段階評価しかない。また構文解析関連の研究として、Collins[2] やShenら [13] の構文解析木のリランキングがあるが、これらは別の手法で構文木の候補が生成されることを仮定している。我々の手法はリランキングではないので、この仮定を置いていない。

3 英語依存構造解析

ここでは、山田ら [14] の英語依存構造解析とほぼ同じシステムに対して、優先度学習を適用することによって、成績が向上できることを示す。なお、Penn Treebankのデータを Collins [1] の主辞規則により変換[1]したものを依存構造の正解データとして用いる。学習データはセクション 02–21 の約4万文で、テストデータはセクション 23 の 2,416 文であ

[1]山田寛康氏の変換プログラムを参考にさせていただいた。

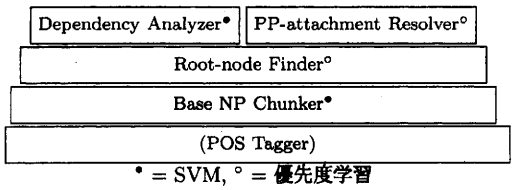

図 2: 英語依存構造解析システムの構造

る。山田らに従い、以下の精度を調べる。ただし、句読点は採点対象に含めないので、テスト中の全単語数は 49,892 語となる。

- DA (dependency accuracy) = 係り先の正解数/全単語数
- RA (root accuracy) = ルートが正しく判定できた文の数/全文数
- CR (complete rate) = すべての係り受けが正しい文の数/全文数

なお、テスト時の品詞は、Collins のパーザの出力に含まれる品詞を利用した。これは Ratnaparkhi のタガー [12] の出力とされている。

我々のシステムは図 2 に示す階層構造になっている。最初に品詞タグ付け器があり、その上に基本名詞句のチャンカーがある。その上にルートノード判定器 (RNF: root-node finder) があり、その上に依存構想解析器と前置詞句係り先判定器 (PPAR: PP-attachment Resolver) がある。基本名詞句を導入したのは、タグ付けが簡単で、精度向上の効果が期待できると考えたためである。基本名詞句のタグ付けについては、CoNLL shared task の定義に従い、SVM に用いて IOB2, one vs. rest [10] で文末から解析するシステムを作成した。

RNF は、山田らの依存構造解析器のルート精度が、Collins や Charniak などの構文解析器に比べて低いので導入したものである。山田らの方法では、ボトムアップに係り受け解析を繰り返すことで係り受けの解析を行う。そのため、ルートに達したときには、かなり精度が下がっている。RNF は、同一文中の各単語の、ルートノードとしての優先度を比較して、優先度が最大の単語をルートノードと判定する。

一方、PPAR は、依存構造解析において、前置詞の係り先が大きな問題となるので導入した。こちらは、各前置詞に対して、同一文中の他の単語の「係り先」としての優先度を比較して、最大になるものを係り先として判定する。

以下では、これらについて説明する。

3.1 ルートノード判定器

RNF は、品詞タグと基本名詞句タグ (B, I, O) の付与された文を入力とする。各単語について、その左右にある単語、品詞タグ、基本名詞句タグ (B, I, O) を bag of words として扱い、素性とする。つまり、「注目している単語の左側に "George" という単語がある」、「右側に品詞 VBG がある」などが素性となる。なお、品詞タグは細かすぎると思われるので、動詞、名詞、形容詞という粗い品詞タグも付与する。さらに、近接する単語は重要であると考え、前後 1 語の単語、品詞タグ、基本名詞句タグについては、別の素性としても扱う。つまり、「すぐ左の単語が "have" である」などの素性を使う。動詞や助動詞のない、名詞句だけの文などもあるので、文中に動詞や助動詞があるか否かも素性とする。20 回以上出現する素性に限り、2 次の多項式カーネルを利用している。図 3 に、SVM と優先度学習の成績を示す。横軸 C はソフトマージンのパラメタである。このグラフによれば、優先度学習の方が SVM より成績が良い。ただし、その差はわずかである。表 1 は、RNF を用いた場合と用いなかった場合の成績の差を示している。これによれば、ルート精度は大きく改善されているが、係り受け精度はわずかに改善されただけである。ただし、文正解率の改善はそれに比べ大きい。

3.2 依存構造解析器

依存構造解析は、山田らの方法をほぼ踏襲している。つまり、品詞タグ付きの文 T を読み込み、文末から開始して各単語を以下の 3 種類に分類する。

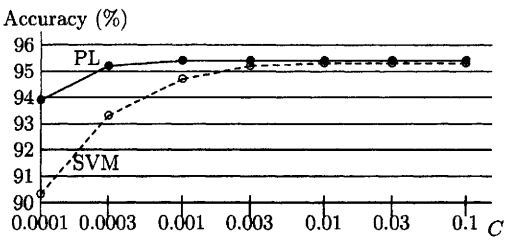

図 3: SVM と優先度学習によるルート精度比較 (トレーニングデータは 1 万文。品詞と基本名詞句は正解を利用。)

表 1: ルートノード検出器の効果
依存構造解析器は 1 万文で学習。ルートノード検出器 (RNF) は全トレーニングデータで学習。DA: 係り受け精度, RA: ルート精度, CR: 文正解率

	DA	RA	CR
RNF なし	89.4%	91.9%	34.7%
RNF あり	**89.6%**	**95.7%**	**35.7%**

- Right: 文中の他の単語がその単語を修飾しておらず、しかも、その単語がすぐ右の単語に係る。
- Left: 文中の他の単語がその単語を修飾しておらず、しかも、その単語がすぐ左の単語に係る。
- Shift: それ以外の場合。

Right の場合、その単語は右の単語の子ノードとなり、Tから削除される。Left の場合、その単語は左の単語の子ノードとなり、Tから削除される。Shift の場合、何もしないで左の単語に移る。

これら 3 つのクラスへの分類なので、one vs. rest により分類する。つまり、それぞれのクラスについて、そのクラスを正例、それ以外のクラスを負例とする分類器を SVM で作っておき、単語の素性ベクトルに対する出力が最大のクラスにその単語を分類する。

なお、山田らの手法では、文中のすべての単語が Shift と判断されると、たとえ複数の単語がTに残っていても、そこで処理を打ち切っていた。ここでは、以下のようにして強制的に処理を再開する。

Tに含まれる各単語に対して以下の量を計算する。

- $\Delta_{\mathrm{Left}}(\boldsymbol{x}) = f_{\mathrm{Shift}}(\boldsymbol{x}) - f_{\mathrm{Left}}(\boldsymbol{x})$,
- $\Delta_{\mathrm{Right}}(\boldsymbol{x}) = f_{\mathrm{Shift}}(\boldsymbol{x}) - f_{\mathrm{Right}}(\boldsymbol{x})$.

たとえば、$\Delta_{\mathrm{Left}}(\boldsymbol{x}) \simeq 0$は、Left の値が Shift の値にきわめて近かったことを表す。そこで、この差が一番小さかった単語を、Shift のかわりにその一番近かったクラスに分類しなおす。これによって単語が一つ減るので、そこから処理を再開できる。

なお、学習データの数が膨大になるので、名詞、動詞、形容詞、前置詞、句読点、その他の 6 つのグループに分割して学習を行なう。

学習には二次の多項式カーネルを用いる。また山田らの素性に加えて、以下のような素性を利用する。

- その単語に対する基本名詞句の B, I, O タグ
- その単語が括弧の中か、引用符の中か
- その単語がルートノードのどちら側か
- その単語に基本名詞句が隣接している場合、その基本名詞句のさらに隣の単語

3.3 前置詞係り先判定器

英語依存構造解析では、前置詞句の係り先が問題となり、PP-attachment と呼ばれている。本稿では、前置詞の係り先の学習を以下のように行なう。まず、各文から前置詞を抜き出す。そして各前置詞に対して同一文中の他の単語を係り先候補とする。SVM の学習では、$(y_i, \boldsymbol{x}_i)$ の $\boldsymbol{x}_i$ として各候補単語の特徴ベクトルを用い、y_i としてその単語が正しい係り先のとき +1、不正解のとき −1 を与える。優先度学習においては、$(y_i, \boldsymbol{x}_{i.1}, \boldsymbol{x}_{i.2})$ の y_i を +1 で固定し、$\boldsymbol{x}_{i.1}$ として正解の係り先単語のベクトルを、$\boldsymbol{x}_{i.2}$ には不正解の係り先単語のベクトルを用いる。この手法は、前置詞に限らずどの品詞でも可能であるが、今回は時間的な制約により、一番問題となる前置詞だけを対象とした。

素性としては、以下のものを用いた。

- 前置詞そのもの: in, of, at など
- その前置詞の前後 2 単語ずつとそれらの品詞
- その前置詞の前の前置詞
- 係り先候補の単語とその品詞
- 前置詞の直後の基本名詞句の最後の単語と品詞
- 前置詞と係り先候補の間の単語の数、コンマの数、前置詞の数、動詞の数、基本名詞句の数、接続詞 (CC) の数
- 前置詞の位置と係り先候補の位置における引用や括弧の深さの差

図 4 に、小規模データによる SVM との比較実験の結果を示す。このグラフによれば、優先度学習の方が SVM よりも成績がよい。ルートのときよりも、はっきりとした差が出ている。さらに、C を変えたとき、SVM の挙動はかなり不安定であるのに対し

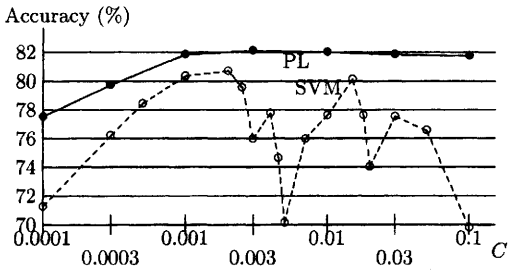

図 4: SVM と優先度学習による前置詞の係り先の精度の比較 (トレーニングデータは 5 千文)

表 2: 前置詞の係り先の精度

Collins3 = Collins' Model 3 parser
PPAR は 3 万 5 千文だけで学習

	IN	TO	average
Collins3	84.6%	87.3%	85.1%
依存構造解析	83.4%	86.1%	83.8%
PPAR	**85.3%**	**87.7%**	**85.7%**

て、優先度学習は安定している。また、3 万 5 千文を使って学習した PPAR の成績と、4 万文を用いて学習した依存構造解析の成績と、Collins の Model 3 パーザの成績を表 2 に示す。これによれば、PPAR の前置詞係り先の精度は依存構造解析の結果のみならず、Collins Model 3 パーザよりもよい。

4 ゼロ代名詞解消

ゼロ代名詞の解消については、磯崎 [7] の手法にほぼ従う。ただし、以下のような変更を行なっている。

- 「が」格だけを対象とする。
- 直前のゼロ代名詞まで正しく解消されていると仮定し、前の失敗をひきずらないようにして評価する。
- 同じ正解候補が複数ある場合、一番近いものだけを残し、他の候補は捨てる。
- 一般記事 General の 30 記事と社説記事 Editorial の 30 記事を合わせた 60 記事で、各記事に対して、残り 59 記事を学習に使い、記事をテストに使う実験を 60 回繰り返す。
- 経験則による順序付けを使わず、単純に $f(\boldsymbol{x})$ (または $g(\boldsymbol{x})$) の成績が最良のものを選ぶ。

表 3: 関連研究との比較

		DA	RA	CR
句情報あり	MEIP	92.1%	95.2%	45.2%
	Collins3	91.5%	95.2%	43.3%
句情報なし	Yamada	90.3%	91.6%	38.4%
	提案手法	**91.2%**	**95.7%**	**40.4%**

- 二次の多項式カーネルを利用。

SVM の学習では、$(y_i, \boldsymbol{x}_i)$ の $\boldsymbol{x}_i$ として各先行詞の特徴ベクトルを用い、y_i としてその先行詞が正解のとき $+1$、不正解のとき -1 を与える。優先度学習においては、$(y_i, \boldsymbol{x}_{i.1}, \boldsymbol{x}_{i.2})$ の y_i を $+1$ で固定し、$\boldsymbol{x}_{i.1}$ として正解の先行詞を、$\boldsymbol{x}_{i.2}$ には不正解の先行詞を用いる。

なお、磯崎ら [7] の結果では、通常の SVM に比べて正例と負例の C を変更した方が成績が良かったので、その結果も示す。ここでは、指定した C を負例の C として採用し、正例の C は以下の式により計算する。

正例の C = 負例の C × 負例の数/正例の数

これをここでは SVMJ と略記する。

二次の多項式カーネルを用い、C を変化させた場合の成績を図 5 に示す。これによれば、やはり優先度学習は SVM よりも良く、SVMJ に比べても良い。飯田ら [6] のトーナメントモデルとの比較はまだ行なっていないが、トーナメントモデルは 2 つの候補のどちらが良いかを学習するもので、アイデア的には優先度学習に似ている。ただし、通常の SVM を用いているので、優先関係の推移律が保証されない問題がある。つまり、候補 A と候補 B を比べて B が選ばれ、候補 B と候補 C を比べて C が選ばれたとしても、候補 A と候補 C を比べた場合に A が選ばれる可能性がある。優先度学習にはそのような問題がない。

5 おわりに

自然言語処理のタスクの中には、複数の候補の中から最良のものを選び出すという作業が必要になるものが多い。本稿では、英語依存構造解析と日本語ゼロ代名詞解消というまったく別のタスクを取り上げ、優先度学習が SVM よりも高精度であることを示した。英語依存構造解析は、前置詞と同様の手法

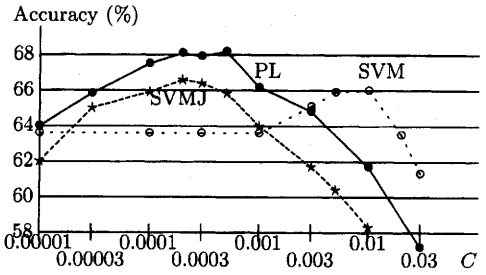

図 5: SVM と優先度学習によるゼロ代名詞解消の精度の比較

で全体を構築することが可能と思われるが、それには、優先度学習に適した高速のアルゴリズムが必要であろう。

日頃討論してくれる知識処理研究グループのメンバーと、研究を支援してくださっている上田部長、管村所長に感謝します。

参考文献

[1] M. Collins. *Head-Driven Statistical Models for Natural Language Parsing*. PhD thesis, Univ. of Pennsylvania, 1999.

[2] M. Collins and N. Duffy. New ranking algorithms for parsing and tagging: Kernels over discrete structures, and the voted perceptron. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 263–270, 2002.

[3] N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines*. Cambridge University Press, 2000.

[4] R. Herbrich, T. Graepel, P. Bollmann-Sdorra, and K. Obermayer. Learning preference relations for information retrieval. In *Proceedings of ICML-98 Workshop on Text Categorization and Machine Learning*, pp. 80–84, 1998.

[5] R. Herbrich, T. Graepel, and K. Obermayer. *Large Margin Rank Boundaries for Ordinal Regression*, chapter 7, pp. 115–132. MIT Press, 2000.

[6] 飯田, 乾, 松本. 文脈的手がかりを考慮した機械学習による日本語ゼロ代名詞の先行詞同定. 情報処理学会論文誌, 45(3):906–918, 2004.

[7] H. Isozaki and T. Hirao. Japanese zero pronoun resolution based on ranking rules and machine learning. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 184–191, 2003.

[8] T. Joachims. Making large-scale support vector machine learning practical. In B. Schölkopf, C. J. C. Burges, and A. J. Smola eds., *Advances in Kernel Methods*, chapter 16, pp. 170–184. MIT Press, 1999.

[9] T. Joachims. Optimizing search engines using clickthrough data. In *Proceedings of the ACM Conference on Knowledge Discovery and Data Mining*, 2002.

[10] T. Kudo and Y. Matsumoto. Chunking with support vector machines. In *Proceedings of NAACL-2001*, pp. 192–199, 2001.

[11] 工藤, 松本. チャンキングの段階適用による日本語係り受け解析. 情報処理学会論文誌, 43(6):1834–1842, 2002.

[12] A. Ratnaparkhi. A maximum entropy part-of-speech tagger. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 1996.

[13] L. Shen and A. K. Joshi. An SVM based voting algorithm with application to parse reranking. In *Proceedings of the Seventh Conference on Natural Language Learning*, pp. 9–16, 2003.

[14] H. Yamada and Y. Matsumoto. Statistical dependency analysis. In *Proceedings of the International Workshop on Parsing Technologies*, pp. 195–206, 2003.