サポートベクトルマシンを用いた中国語解析実験

吉田 辰巳[†]　　大竹 清敬[††]　　山本 和英[††,†††]

現在入手可能な解析器と言語資源を用いて中国語解析を行った場合にどの程度の精度が得られるかを報告する．解析器としては，サポートベクトルマシン (Support Vector Machine) を用いた YamCha を使用し，中国語構文木コーパスとしては，最も一般的な Penn Chinese Treebank を使用した．この両者を組み合わせて，形態素解析と基本句同定解析 (base phrase chunking) の 2 種類の解析実験を行った．形態素解析実験の際には，一般公開されている統計的モデルに基づく形態素解析器 MOZ との比較実験も行った．この結果，YamCha による形態素解析精度は約 88%で MOZ よりも 4%以上高いが，実用的には計算時間に問題があることが分かった．また基本句同定解析精度は約 93%であった．

キーワード:　中国語解析，サポートベクトルマシン，同定解析，YamCha, MOZ

Performance Evaluation of Chinese Analyzers with Support Vector Machines

Tatsumi Yoshida[†] and Kiyonori Ohtake[††] and Kazuhide Yamamoto[††,†††]

We will report performances of currently and publicly available Chinese analyzers and resources. We use YamCha, a tool based on Support Vector Machines, and the Penn Chinese Treebank as a language resource. Combining these two, we measure the performances of Chinese analysis, i.e., word segmentation, part-of-speech tagging, and base phrase chunking. In the experiment of word segmentation and part-of-speech tagging, we also report the performance of MOZ, a statistical morphological analyzer, which is also available to the public. We found that the accuracy of morphological analysis using YamCha attains around 88%, which is over 4% higher than that of MOZ, although it is computationally very expensive. We also found that the accuracy for base phrase chunking is approximately 93%.

KeyWords:　*Chinese analysis, Support Vector Machine, Chunking, YamCha, MOZ*

1　はじめに

自然言語処理を進める上で，形態素解析器をはじめとする言語解析器は，コーパスなどの言語資源と同様に最も重要な道具である．近年では，この重要性は研究者間でほぼ認識されており，英語や日本語に対する形態素解析器と構文解析器はいずれも複数のものが作成，そして公開または市販され，我々研究者はその恩恵に預かっている．

† 豊橋技術科学大学 知識情報工学系, Department of Knowledge-based Information Engineering, Toyohashi University of Technology
†† ATR 音声言語コミュニケーション研究所, ATR Spoken Language Translation Research Laboratories
††† 長岡技術科学大学 電気系, Department of Electrical Engineering, Nagaoka University of Technology

ところが，中国語に関しては以上の状況は同じではない．我々の知る限り，日本国内はもちろん，中国においても誰もが手軽に使える中国語解析器が研究者の間で広範に知られている，という状況にはなく，まだ十分に解析器が整備されているとは言えない．

この背景の一つは，中国語解析の困難性であると考える．中国語は英語のように概ね単語ごとに分かち書きされてはおらず，単語分割が必要である．また，文字種が単語分割のための大きな情報を持つ日本語とは異なり，ほぼ単一文字種 (漢字) である．さらに，複数品詞を持つ語が多いため品詞付与も容易ではない．たとえば，中国語の介詞 (前置詞) のほとんどは動詞からの転成であるため日本語や英語にはほとんど存在しない内容語と機能語との間で品詞付与の曖昧性が生じる．たとえば “到北京了”（北京に着いた）の “到” は動詞（到着する）であるが “到北京去”（北京に行く）の “到” は介詞（· · · に）であり，すなわち “到北京” だけでは “到” の品詞は決定できない．また日本語における「ーする」(動詞)「ーい」(形容詞) などの明確な文法標識を持たないため，内容語間の曖昧性も比較的多い．たとえば中国語の “担心” は日本語の「心配 (名詞)/心配する (動詞)/心配だ (形容詞)」のすべてに相当する．

我々は現在，中日翻訳，並びに中国語換言処理の研究を行っている (張, 山本, 坂本 2002)．これらの処理は中国語が入力であるため，表層処理を行わない限り中国語解析器が必要である．このため我々は，現在入手可能な解析器や言語資源を組み合わせて中国語解析を行うことを試みた．ここで，中国語構文木コーパスとしては，現在一般的な Penn Chinese Treebank (以下，CTB とする) を使用した．一方，解析器としてはサポートベクトルマシン (Support Vector Machine，以下，SVM) に基づく YamCha を使用した．SVM ならびに YamCha については 2.2 節でその概要を述べる．

本報告では，形態素解析と基本句同定解析 (base phrase chunking) の 2 種類を行った．3 節で形態素解析について，4 節で基本句同定解析について述べる．それぞれの解析で，学習文テストと未知文テストの 2 種類の解析精度を測定し，考察を行った．形態素解析実験では，連接コスト最小法に基づく形態素解析器 MOZ を使用して，解析精度の比較を行った．さらに，日本語と比較してどの程度中国語の形態素解析が難しいのかを調べるために京都大学テキストコーパスを用いて実験した．また，品詞タグ付けに限定すれば，CTB よりも大きなコーパスが入手可能であることから，CTB の約 11 倍の大きさを持つ人民日報タグ付きコーパスを用いての形態素解析実験も行った．

本報告の主な目的は，上記の解析器と言語資源を用いて中国語解析器を構築した場合，どの程度の解析精度が得られるのかを報告することにある．すなわち，この解析器にどのような問題がありどのような改善が可能かを提案するという提供者の視点ではなく，使用者の視点，すなわち中国語処理に携わる研究者にとってこの解析器がどの程度有用であり，使用の際にはどのような点に注意が必要か，などを報告することに主眼がある．

いずれも容易に得られるツールと言語資源を組み合わせた場合にどのような精度が得られるかを測定，報告することは誰にでもできる作業である．しかし，研究者が研究の必要性のためできるだけ高精度の解析器を求める状況にある場合，本報告のような報告によって解析の期待

精度を予め知った上で同一の解析器を構築できる．あるいは，研究上より高精度の解析器が必要な場合は最初から別の選択肢を考えることもできる．このように，我々は中国語処理を行う研究者への有益性を考え，我々で測定した解析精度を技術資料として報告することにした．

2　中国語解析のための言語資源と解析環境

本節では，我々の実験で使用した言語資源と解析環境の概略を述べる．

2.1　中国語構文木コーパス

我々は，入手可能な中国語言語資源として，現在最も一般に知られていると考えられる Penn Chinese Treebank (CTB) を使用した．CTB は，米国ペンシルバニア大学 (University of Pennsylvania) の Chinese Treebank Project により作成された構文木コーパスである．このコーパスの概要ならびに入手方法を付録 B.1 に示す．以下に述べる実験では，このプロジェクトの最終版である LDC2000T48 を用いた．また，CTB で定義されている中国語品詞数は 33，句情報の数は 17 である．この一覧を巻末の付録 A に示す．

2.2　SVM による同定解析

SVM は，d 次元の特徴ベクトル (パターン) $\mathbf{x}$ を定められた二つのクラス (A, B) のいずれかに識別する 2 値クラスの線形識別器である．また，SVM では，カーネルトリックと呼ばれる計算技術によって非線形識別器を実現できる．従来の手法と比べて多くの面で優位性を示し，文字認識や画像認識など，様々な分野で応用されている．

識別器は，識別関数 $f(\mathbf{x})$ の形によって与えられ，f が正ならクラス A, f が負ならクラス B に識別される．$f(\mathbf{x}) = 0$ を満たす $\mathbf{x}$ の集合を識別面と呼ぶ．SVM の大きな特徴の一つは，マージン最大化である．マージンとは，識別面と特徴ベクトル間の最小距離であり，マージンが大きいほうが，汎化能力が高く，テストパターンを精度良く識別できる．一般に，学習パターンを識別する超平面は複数存在する．SVM では，上に述べた理由から超平面と学習パターンとの最小距離を最大にする超平面を求め，これを識別面とする．

決定した超平面からの最小距離に対応する特徴ベクトルをサポートベクトルと呼ぶ．またサポートベクトル以外の特徴ベクトルは最終的に得られる識別関数に一切影響を及ぼさない．したがって，出現頻度などの統計量を用いる識別器 (たとえば決定木など) とは性質が異なる．

SVM を用いることの短所は，(1) 2 値クラス識別器であるため多クラスを考慮に入れた識別関数の最適化ができない (2) 計算量が大きい (3) 問題に適したカーネルトリック (カーネル関数) の明確な選択方法は知られていない，などである (前田 2001)．

自然言語処理における同定解析 (chunking) とは，与えられた言語的な要素列 (文字列，単語

列など) をより上位概念の言語的要素 (単語，句，文など) にまとめあげるために，各要素に情報を付与する一連の処理を指す．たとえば，単語の分かち書きや形態素解析，文節まとめあげ，テキストセグメンテーション，文書分類などはすべて同定解析とみなすことができる．

工藤は，SVM に基づく汎用的な同定解析器として YamCha (Yet Another Multipurpose CHunk Annotator) [1] を公開している．YamCha は同定解析を各要素に対する情報付与と見なすため，一般的な解析器として用いることが可能である．

SVM は 2 値識別器であるため，情報付与 (tagging) のような多値クラスの識別問題を扱うためには，何らかの拡張を行う必要がある．これに対して YamCha では，*pairwise classification*(Kreßel 1999) （一対比較分類）と呼ばれる手法を採用している．これは，K クラスの識別問題を解くために，各クラス 2 つの組み合わせを識別する $K \times (K-1)/2$ 種類の識別器を作成し，最終的にそれらの多数決でクラスを決定する手法である．

SVM を用いた自然言語解析の例として英文の基本句同定実験 (工藤, 松本 2000) や，日本語の係り受け解析実験 (工藤, 松本 2002) があり，従来手法と比較して高い解析結果を示している．また，平と春野は，SVM を用いた文書分類について，高い分類精度を得るためには品詞によるフィルタリングをした後，全単語を入力として用いればよいことを示している (平, 春野 2000)．

3　YamCha による形態素解析

SVM を用いた中国語の解析器として，我々は YamCha を用いた．この節では YamCha による中国語の形態素解析について述べる．

形態素解析を文字のならびを形態素へまとめあげる同定解析と見なす．したがって，各文字がチャンク (chunk) を構成する 1 要素に相当する．チャンクとは，同定解析における同定単位を指し，ここでは形態素に相当する．SVM の学習のために CTB を正解データとして用いる．

3.1　YamCha の準備

YamCha で扱うデータ形式は，複数のトークンと複数のカラムから構成される．各行は入力のトークンに対応する．形態素解析を行う場合は，1 トークンが 1 文字に対応する．各カラムにはトークンに付与された属性が記述される．また，各カラムはタブまたはスペースによって区切られている必要がある．YamCha によって推定 (学習) すべき属性は最後のカラムに与える．ここでは，形態素解析を行うので，第 1 カラムには，形態素の要素である 1 文字を記述し，第 2 カラムには，YamCha で推定する情報を記述する．この情報には，形態素の区切り位置を示す情報と，形態素に付与する品詞情報の両方が含まれる．また，文と文の境界は，EOS と記述した行，もしくは空行を付与することで同定する．

トークンがチャンクに含まれるか否かの状態を示すために IOB2 モデルを用いた (Sang and

[1] http://cl.aist-nara.ac.jp/~taku-ku/software/yamcha/

Veenstra 1999)．これは，あるトークンがチャンクの先頭ならば B タグを付与し，チャンクに含まれる先頭以外のトークンならば I タグを付与し，チャンクに含まれない場合には O タグを付与するモデルである．一方，本実験ではすべてのトークンが何らかのチャンクに含まれるため O タグは用いられない．

付与する品詞タグセットは CTB のタグセットと同一である．また，CTB において品詞が “-NONE-” の形態素は構文構造上形式的に配置され，実体を持たないため対象外とする．最終的にトークンに付与されるタグは B/I タグと品詞タグを “–” で結んだものとなる．CTB から YamCha で中国語形態素解析を行うための書式へ変換する概要を図 1 に示す．

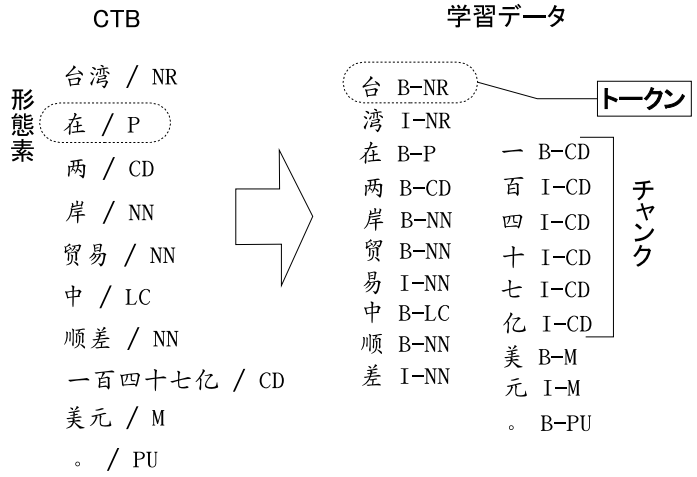

図 1: YamCha 用中国語形態素解析データ書式

以上の処理で得られた学習データを YamCha に与え，SVM のモデルを作成する．その際に，素性として使用したデータは YamCha の標準設定に従った．すなわち，推定するトークンと，その前方，および後方 2 トークンの計 5 トークンにおける文字データと前方 2 トークンの推定タグを素性として学習した．解析方向は前方からである．これは，使用する素性を変化させた場合の精度を検討した予備実験の結果において，YamCha の標準設定が最も高い精度であったためである．これらの関係を図 2 に示す．また，YamCha で学習を行うために用いた SVM の実装は同じく工藤が公開している TinySVM 0.08 [2] である．

[2] http://cl.aist-nara.ac.jp/~taku-ku/software/TinySVM/

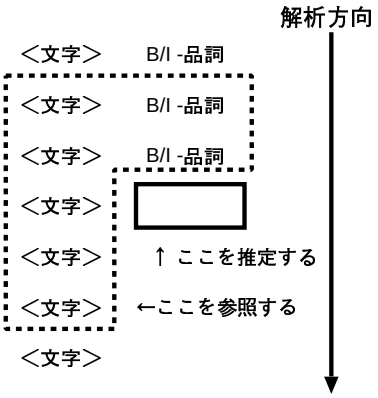

図 2: 学習素性

3.2　形態素解析器 MOZ

本報告では，YamCha と同程度の時間的コストで実現できる中国語形態素解析器として MOZ[3] (山下, 松本 2000) をとりあげ，両者の比較を行う．本節では，MOZ に関する概略を述べる．

MOZ で形態素解析を行うためには，形態素辞書と接続表が必要となる．MOZ はコスト最小法に基づく解析器であるので，形態素辞書と接続表にはそれぞれコストを与えなければならない．ここでは，CTB から得られる情報 (品詞 2 つ組の頻度や形態素の頻度など) から形態素辞書と接続表，ならびにそれらのコストを求める．すなわち，形態素辞書は形態素とその出現確率から，品詞接続表は品詞 bi-gram によって与える．MOZ では，品詞接続表に tri-gram 以上のデータを用いることができるが，データ過疎性 (data sparseness) による精度低下を避けるために本実験では品詞 bi-gram のみを用いた．

形態素を w_i，品詞を POS_i，x の頻度を $C(x)$ と表記すると品詞が POS_i である形態素 w_i の出現確率を式 (1) で与える．ここで，$C(w_i,\ POS_i)$ は形態素 w_i，かつその品詞が POS_i である頻度を示している．

$$p(w_i \mid POS_i) = \frac{C(w_i,\ POS_i)}{C(POS_i)} \tag{1}$$

また，品詞接続表の確率は式 (2) で与える．ここで $C(POS_i,\ POS_j)$ は品詞 POS_i のあとに品詞 POS_j が出現した頻度である．

$$p(POS_j \mid POS_i) = \frac{C(POS_i,\ POS_j)}{C(POS_i)} \tag{2}$$

システムで扱う最高コストを 128 として，コスト化係数を求める．コスト化係数は式 (3) により与えられる．ここで最小確率は，すべての $p(w_i \mid POS_i)$ および $p(POS_j \mid POS_i)$ における

[3] http://cl.aist-nara.ac.jp/student/tatuo-y/ma/

最小値である．

$$\text{コスト化係数} = |\ \text{最高コスト} / \log(\text{最小確率})| \tag{3}$$

形態素辞書ならびに接続表のコストはそれぞれの確率から式 (4) により与えられる．

$$\text{コスト} = \lceil |\log(\text{確率}) \times \text{コスト化係数}\ | \rceil \tag{4}$$

以上述べた方法により形態素辞書ならびに接続表のコストを計算する．

3.3 学習文テスト

まず，CTB 全体を学習データ (4181 文[4]) とし，この中から無作為に抽出した 1 割の文（418 文） を解析する学習文テスト (closed test) を行った．具体的には，YamCha と MOZ をそれぞれ用いて，418 文からなるテストデータを解析し，その結果を CTB の正解と比較し，評価した．その結果から，再現率 (recall) と適合率 (precision) を算出した．再現率と適合率はそれぞれ式 (5) および式 (6) とした．

$$\text{再現率} = \text{解析結果中の正解形態素数}/\text{正解形態素数} \tag{5}$$

$$\text{適合率} = \text{解析結果中の正解形態素数}/\text{解析結果の形態素数} \tag{6}$$

再現率と適合率から F 値 (F-measure) も求めた．F 値は再現率 R と適合率 P の調和平均であり，式 (7) によって与えられる．

$$F\ \text{値} = \frac{2 \times R \times P}{R + P} \tag{7}$$

ただし，本実験では正解形態素数を求める場合に形態素分割のみ正解の場合と，分割ならびに品詞の両方の 2 段階の条件を設けて評価した．この結果を表 1 に示す．また，品詞誤りの上位 10 件を表 2 に示す．ここで，出現率とは誤りの総数に対する各誤りの割合を示す．

3.4 未知文テスト

次に，CTB 全体を母集団とする 10 分割交差検定 (cross validation) による未知文テスト (open test) を行った．

まず，CTB 全体 (4181 文) を無作為に 10 等分し，1 割をテストデータ，残りの 9 割を学習データとする．この方法で 10 組の学習データとテストデータを作成した．YamCha と MOZ そ

[4] CTB の説明には 4185 文とあるが，我々が発見した明らかな誤り，たとえば句点のみを 1 文とするなど，を除くと 4181 文となった．

表 1: 学習文テストの正解率

		分割のみ			分割と品詞付与		
対象	解析器	再現率	適合率	F 値	再現率	適合率	F 値
CTB	YamCha	99.91%	99.93%	99.92%	99.58%	99.60%	99.59%
	MOZ	97.78%	98.82%	98.82%	93.74%	94.73%	94.23%
PKU	YamCha	99.95%	99.94%	99.94%	99.76%	99.75%	99.75%
	MOZ	98.72%	99.17%	98.94%	94.71%	95.14%	94.92%

表 2: 学習文テスト (CTB) における品詞誤りの上位 10 件

YamCha		MOZ	
正解 - 解析	出現率	正解 - 解析	出現率
NN - NR	14/33	VV - NN	84/401
NR - NN	6/33	DEC - DEG	78/401
VV - NN	5/33	NN - VV	38/401
AD - JJ	2/33	NN - NR	19/401
AD - NN	1/33	VV - P	14/401
NN - VV	1/33	CC - AD	13/401
JJ - AD	1/33	NN - JJ	12/401
CD - OD	1/33	JJ - NN	10/401
SP - DEC	1/33	AD - JJ	9/401
DEC - DEG	1/33	JJ - AD	7/401
— - —	—	P - AD	7/401

れぞれに対して，10 組の学習データとテストデータを用いて学習ならびにテストを行い，平均値を求めた．

実験結果を表 3 に示す．また，品詞誤りの上位 10 件の内訳を表 4 に示す．表中の null は未知語のために付与されたタグを示している．

未知文テストにおいて YamCha が 1 学習データを学習するために要した処理時間と 1 テストデータを解析するために要した処理時間などを表 5 に示す．測定時は，CPU: Pentium III 600 MHz，メモリ: 256 MB，OS: Linux の計算機を用いた．ただし，YamCha で解析を行うためには，アーキテクチャ非依存のテキスト形式のモデルファイル (学習結果を格納するファイル) をアーキテクチャ依存のバイナリ形式にコンパイルする必要がある．その際にテキスト形式のモデルをメモリ上に展開するため大量のメモリを必要とする．この実験では，約 650 MB のメモリを必要としたため，コンパイル作業だけは，CPU: Pentium III 733 MHz，メモリ: 960

表 3: 未知文テスト結果

		分割のみ			分割と品詞付与		
対象	解析器	再現率	適合率	F 値	再現率	適合率	F 値
CTB	YamCha	93.04%	93.71%	93.37%	87.58%	88.20%	87.89%
	MOZ	92.19%	85.89%	88.93%	86.32%	80.42%	83.26%
京大	YamCha	92.02%	93.23%	92.62%	88.17%	89.33%	88.74%
(10 万語)	JUMAN	98.97%	98.65%	98.80%	93.49%	93.19%	93.34%
PKU	YamCha	86.66%	87.52%	87.09%	80.19%	80.99%	80.59%
(10 万語)	MOZ	90.05%	80.58%	85.05%	84.57%	75.67%	79.87%
PKU	YamCha	95.19%	95.19%	95.19%	91.72%	91.72%	91.72%
	MOZ	95.68%	93.42%	94.58%	89.87%	87.75%	88.72%

表 4: 未知文テストにおける誤り品詞の上位 10 件

YamCha		MOZ	
正解 - 解析	出現率	正解 - 解析	出現率
VV - NN	1037/5447	VV - NN	1016/5859
NN - VV	579/5447	DEC - DEG	933/5859
DEC - DEG	578/5447	NN - VV	701/5859
JJ - NN	402/5447	NN - NR	263/5859
DEG - DEC	334/5447	DEG - DEC	186/5859
NR - NN	323/5447	VV - P	184/5859
NN - NR	266/5447	NN - JJ	126/5859
VA - NN	174/5447	JJ - AD	118/5859
AD - NN	126/5447	NN - null	116/5859
AD - VV	115/5447	CC - AD	100/5859

MB の計算機を使用した．このコンパイルに要した時間は約 5 分であった．

本実験のあと，コンパイルに必要なメモリ量を抑える目的からプログラムを修正した．その結果，速度を少々犠牲にするが，80MB 程度のメモリで，上記のモデルファイルをコンパイル可能となった．処理時間は，CPU: Pentium III 600MHz，メモリ: 256 MB の計算機で約 7 分であった．この YamCha 0.1 に対するプログラムの差分は WWW ページ[5] にて公開している[6]．

一方，MOZ が学習データ (約 3700 文) から形態素辞書と接続表のコストを求めるために必

[5] `http://www.slt.atr.co.jp/~kohtake/`

[6] 2002 年 11 月に公開された YamCha 0.2 では，この修正が反映されておりプログラムの差分を適用する必要はない．

表 5: YamCha での処理時間

	学習	解析
タグの種類	53	—
文数	約 3700	約 400
トークン数	約 15 万 8 千	約 1 万 7 千
処理時間	約 6 時間	約 35 分

要とした時間は約 3 秒であり，1 テストデータ (約 400 文) を解析するために必要とした時間は約 1 秒であった．

3.5 未知語の性質

次に，テストデータに含まれる形態素のうち，学習データに含まれていないものを未知語と定義し，その性質を調べた．未知文テストにおける平均未知語率などを求めた．結果を表 6 に示す．未知語率とはテストデータの単語数に占める未知語数の割合を指し，平均未知語率とはテストセット全体での未知語率の平均を示している．

表 6: 未知文テストにおける平均未知語率など

	平均未知語率			形態素数	
対象	[異なり/のべ](%)	平均文長	平均形態素長	異なり	のべ
CTB (10 分割)	21.74 / 7.05	41.10 字	1.72 字	12079	103901
京大 10 万語 (10 分割)	24.18/ 8.38	43.91 字	1.77 字	14613	102310
PKU10 万語 (10 分割)	26.85 / 10.87	40.71 字	1.64 字	16810	102741
PKU (11 分割)	15.79 / 2.84	41.85 字	1.64 字	61846	1118794

未知語に対する解析器の性質をより詳しく調べるため，以下の実験を行った．まず，CTB から 40 記事を無作為に選択する．そのうち 30 記事を学習データ，残り 10 記事をテストデータとする．次にテストデータから順に 1 記事ずつ学習データに加えていき，計 11 個の学習データを作成する．それぞれの学習データに基づく解析器で同一のテストデータ 10 記事を解析した．以上の実験概要を図 3 に示す．

テストデータに含まれる単語数は 1111，のべ語数は 2954 である．11 のテストセットにおける学習データの単語，未知語数ならびに未知語率を表 7 に示す．

各テストセットにおける未知語率 (異なり) と解析精度の関係を図 4 に示す．図では，解析精度を F 値で示す．なお，未知語率 (のべ) と精度の関係を図示していないが，図 4 とほぼ同一の図となるため省略する．

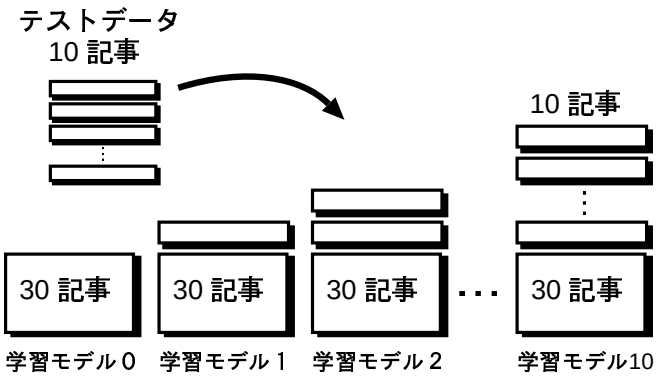

図 3: 未知語と解析精度に関する実験の概要

表 7: 未知語と解析精度に関する実験におけるテストセットの単語数

	単語数		未知語数		未知語率 (%)	
テストセット	異なり	のべ	異なり	のべ	異なり	のべ
0	2809	10819	572	770	51.49	26.07
1	2892	11402	489	639	44.01	21.63
2	2921	11512	460	594	41.40	20.11
3	3044	11929	337	442	30.33	14.96
4	3075	12134	306	406	27.54	13.74
5	3127	12402	254	337	22.86	11.41
6	3190	12764	191	252	17.19	8.53
7	3265	12993	116	150	10.44	5.08
8	3293	13274	88	115	7.92	3.89
9	3319	13393	62	75	5.58	2.54
10	3381	13773	0	0	0.00	0.00

次に，未知語がある場合の解析結果を調査した．テストセット t_i における未知語の集合を $UK(t_i)$，$w \in UK(t_i)$ のうち形態素分割に成功した形態素の集合を $USeg(t_i)$ とする．さらに，$w \in USeg(t_i)$ のうち品詞も正しく解析された形態素の集合を $USP(t_i)$ とする．これらの集計結果を表 8 に示す．また，MOZ では入力に未知語が含まれる場合，解析不能で停止することはないが，最終的に未知語と判断された文字列を 1 文字ずつ，null という品詞を与えて出力する．したがって，MOZ での解析で $|USP(t_i)|$ を示していないのは，MOZ では未知語が null と解析されるため，$USP(t_i)$ は空集合となるためである．一方，MOZ での解析において $USeg(t_i)$ が得られるのは，正解が 1 文字の形態素である場合に形態素分割が成功したと見なすからである．

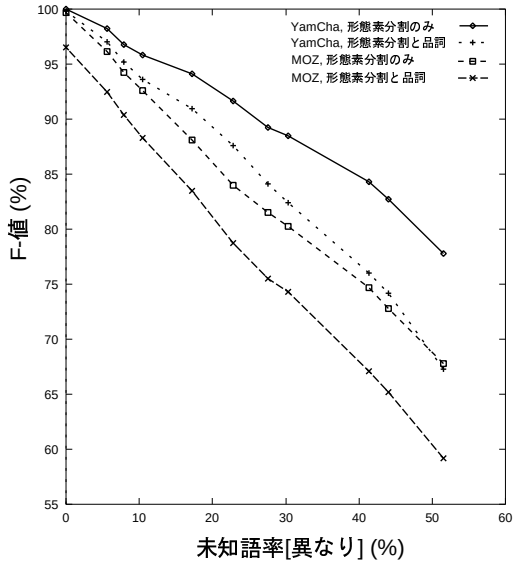

図 4: 未知語率 [異なり] と解析精度

表 8: 未知語とその解析結果

		YamCha		MOZ
t_i	$\|UK(t_i)\|$	$\|USeg(t_i)\|$	$\|USP(t_i)\|$	$\|USeg(t_i)\|$
0	572	376	249	87
1	489	319	212	78
2	460	299	199	77
3	337	225	157	56
4	306	202	147	47
5	254	174	127	35
6	191	135	95	27
7	116	82	59	18
8	88	60	43	15
9	62	47	35	10

3.6　言語依存性とコーパスの大きさ

これまで，CTB をコーパスとして，中国語の形態素解析について 2 つの解析器，YamCha と MOZ を比較してきた．しかしその未知文テストの結果は，表 3 に示す通り，これまでに報告されている日本語の形態素解析器の精度より低い．ここでは，その原因が，中国語の言語と

しての解析の難しさにあるのか，コーパスの量にあるのかを検討する．

3.6.1　日本語形態素解析における YamCha

CTB で用いられているのは，新華社通信の新聞記事である．そこで，SVM に基づく形態素解析器の日本語に対する精度を検証するために我々は，京都大学テキストコーパス 第 3.0 版（以下，京大コーパスと呼ぶ）を用いて実験を行った．このコーパスの詳細については付録 B.3 を参照されたい．

CTB の大きさが約 10 万語であるところから，我々は，京大コーパスのうち 1 月 1, 3, 4, 5 日の記事，4117 文，102310 単語を用いることにした．CTB 全体では，4181 文，99720 単語である．

我々が選択した京大コーパスの一部について CTB に対する実験と同様に，10 分割交差検定を実施した．この検定における平均未知語率などを表 6 に示す．

京大コーパスを用いた実験における品詞は，JUMAN[7]が定義する品詞のうち品詞細分類までを含めたものとした．この結果，タグセットの大きさは，41 となり，CTB の 33 より大きい．YamCha の学習に用いたデータの例を以下に示す．

```
今 B-名詞-時相名詞　　話 B-名詞-普通名詞
年 I-名詞-時相名詞　　題 I-名詞-普通名詞
の B-助詞-接続助詞　　の B-助詞-接続助詞
大 B-名詞-普通名詞　　力 B-名詞-普通名詞
相 I-名詞-普通名詞　　士 I-名詞-普通名詞
撲 I-名詞-普通名詞　　た B-接尾辞-名詞性名詞接尾辞
を B-助詞-格助詞　　ち I-接尾辞-名詞性名詞接尾辞
支 B-動詞
え I-動詞
る I-動詞
```

実験結果を表 3 に示す．参考までに我々が選択した京大コーパスの一部を JUMAN で形態素解析した結果もあわせて示す．

日本語を対象とした実験でも，同程度の大きさのコーパスでは，同程度の精度となった．

3.6.2　コーパスの大きさと解析精度

CTB は既に述べた通り 4181 文，99720 単語からなるコーパスだが，大きいとは言えない．そのため，10 分割交差検定を行っても，テストセットにおける未知語率が非常に大きくなり，

[7] http://www-nagao.kuee.kyoto-u.ac.jp/nl-resource/juman.html

精度が低くなる．

品詞タグ付けされたコーパスがあれば，それを形態素解析器のために利用することが可能である．CTB のように構文木を備えている必要はない．品詞タグ付けされた中国語のコーパスはいくつかあるが，我々は，CTB よりも大きく，そして同じ新聞記事という点から人民日報タグ付きコーパスを使用した．人民日報タグ付きコーパスに関しては，付録 B.2 にその概要等を示す．ただし，我々は人民日報半年分であるコーパス全てではなく，無償公開している 1ヶ月分のデータを用いた．人民日報タグ付きコーパス 1ヶ月分 (以下，PKU と呼ぶ) は，44011 文，1121447 単語からなるコーパスである．定義されているタグセット[8]の大きさは 39 である．しかしながら，実際の PKU にはここに定義されていないタグが 7 種類 (Bg: 8, Mg: 7, Rg: 10, Yg: 1, na: 1, nx: 459, vvn: 1，コロンの後の数値は頻度を示す) 出現する．ここで，nx は，定義されているタグ x に該当することがわかったので，nx 以外の 6 種類のタグを含む文を除いた．その結果，PKU は，43913 文，1118794 単語からなるコーパスとなった．

PKU は CTB の約 11 倍の大きさを持つ．まず，同程度のコーパスの大きさでの精度を検証した．PKU をランダムに文単位で 11 等分し，そのうちの 1 つ (約 10 万語) を用いて 10 分割交差検定を行った．結果を表 3 に示す．以下，この PKU の 10 万語のコーパスを PKU10 万語と表記する．

PKU10 万語を用いての未知文テストと CTB での未知文テストでの精度の違いは，両者のコーパスの違いに起因する．両者はともに約 10 万語のコーパスであるが，表 6 から，PKU の方が未知語が多く，なおかつ 1 文あたりの平均形態素数が CTB より約 1 単語多いことがわかる（CTB での 1 文平均形態素数 = 41.10/1.72 = 23.90，PKU10 万語での 1 文平均形態素数 = 40.71/1.64 = 24.82）．さらに，PKU は，CTB と比較してタグセットが大きく，タグ一種類あたりの学習データが少なくなることからタグの推定がより難しくなっている．したがって，PKU10 万語での結果は，同じ 10 万語の CTB と比較して精度が大きく低下したと考える．

MOZ が再現率の面で YamCha を上回るのは，辞書を用いる利点が活かされていると考える．YamCha は 1 文字単位でタグの推定を行う．タグの推定に用いるのは推定対象文字とその前後 2 文字，さらに直前に推定した 2 つのタグの計 7 つの素性である．したがって，学習量が不十分な状態では，ある形態素が学習，テストコーパスの両方に含まれている場合でも，テストデータにおける当該形態素とその周辺文字列の組合せを素性として学習している可能性は低いため SVM が誤る可能性が大きくなる．一方，MOZ では，一度辞書に登録された形態素は，形態素分割および品詞の曖昧性が生じない限り正しく再現される．さらに学習コーパスが大きくない場合では，これらの曖昧性が発生する頻度は低いと予想する．したがって MOZ が再現率の面で，YamCha を上回ったと考える．

次に，PKU 全てを学習コーパスとし，学習文テストを行った．テストデータとして PKU から無作為に抽出した 3993 文，101218 単語を解析した．なお，学習に用いるコーパスが大きくなることから，より大きな分解能が必要になると考え，MOZ のコスト化係数を 128 から 1024

[8] http://www.icl.pku.edu.cn/research/corpus/addition.htm

へと変更した．実験結果を表 1 に示す．この結果から，学習コーパスが 100 万語を越えても，YamCha は変わらず高い性能を示していることがわかる．

学習に用いるコーパスの大きさが非常に大きくなった場合の 2 つの解析器のふるまいを検討するために，11 等分したデータを用いて 11 分割交差検定を行った．結果を表 3 に示す．

3.7　形態素解析結果に関する考察

以上得られた形態素解析に関する実験結果について考察する．

まず，YamCha と CTB を使用した場合の未知文に対する形態素解析 (分割と品詞付与) 精度 (F 値) は 87.9%であった．同条件で MOZ が 83.3%であることを考えると，言語資源として CTB しか得られない条件下では YamCha を使用したほうが高精度な解析器を実現できる．

次に，解析時間については YamCha が極端に遅い．学習時間も同様である．よって解析時に実時間性を問われる状況においては MOZ を使用すべきである．YamCha では，既に述べたように，一対比較分類に基づき品詞付与を行うため，品詞数が大きくなると，その 2 乗に比例する SVM が必要となる．そのため，品詞数の増加とともに，学習，解析時間が増大する．

品詞付与誤りの傾向では，1 節で述べたように中国語において本質的に解析の難しいと予想される箇所で両解析器共に誤っており，解析器としての誤りのくせはあまり見受けられない．

未知語に対する頑健性については，YamCha のほうが優れる．実験では，YamCha は未知語の約 4 割を正しく解析しており，頑健性を確かめられた．この割合は，未知語率が変化しても，大きく変化することはなく，実験した範囲の未知語率 (51.5%から 6.4%) で，40%から 45%程度であった．このことから，未知語率が大きくなったからといってそれに影響されて極端に精度が低下することはないと予想する．一方，MOZ は，未知語に対して 1 文字ずつに null という品詞を付与して出力するのみであるため，何らかの拡張を行わない限り品詞の推定を行えない．したがって，再現率に対して適合率が低くなる傾向がある．また，YamCha にはこのような傾向はなく，適合率が再現率を若干上回る傾向を示す．これらのことから，入力文中に多くの未知語の存在が予想される場合，あるいは学習データの語彙傾向と異なる入力文を解析する場合は YamCha を用いたほうがよい．

ただし，一般的な状況としてコーパスとは別個に単語集合を入手できる場合がある．この場合には MOZ を使うべきだろう．YamCha では単語集合があってもこの情報を学習に反映させることができず，コーパス中の出現単語のみが学習対象であるためである．

言語資源をより活用しているのは YamCha であるが，辞書を用いないことから語彙的整備ができない．また人間の内省による知見を反映させにくい．したがって，既に大量のタグ付きコーパスが存在する状況では，MOZ のような，接続コストを統計的言語モデルに基づいて推定する手法が頑健で，整備しやすい解析器となる．逆に，タグ付きコーパスが充分に整備されていない言語の解析器を必要とする場合，あるいは，新たに定義した品詞に対する解析器が，その品詞で解析されたコーパスが充分に存在しない状況で必要となる場合には，YamCha が有効

である．

また，中国語に固有の解析の難しさが考えられるが，日本語を対象とした実験の結果から，同程度のコーパスならびにそのタグセットの大きさの場合では顕著な違いは見られなかった（表3）．しかしながら，京大コーパスの平均未知語率が，CTB のそれと比較して大きいことと（表6），京大コーパスのタグセット（41）が CTB のそれ（33）より大きい．これは，日本語解析の実験条件が中国語解析の条件に比べ，厳しいことを示す．それにもかかわらず，実験結果は同程度の精度を示した．これらのことから中国語解析が，日本語解析に比べて難しいと判断する．さらに，表3は，京大コーパスの解析結果と CTB の解析結果において単語分割のみと分割と品詞付与との間の逆転現象があることを示している．これは，中国語解析の困難な点は，品詞付与にあるという我々の予見を裏付ける結果と考える．

一方で，より大きなコーパスを用いることにより，より高精度な解析器が実現可能であることが，表3からわかる．また，表1に示した学習文テストの結果から，学習コーパスをさらに大きくすると YamCha はさらに精度を向上させる可能性がある．それに対し，MOZ は，PKU（100 万語以上の大きさを持つコーパス）を用いての学習文テスト結果において分割と品詞付与の F 値が約 95%（表1）だったことから，現状の枠組のままでは，F 値で 95%程度がその性能の限界だと考える．これをさらに向上させるためには，接続表への tri-gram 規則の適用ならびにその補完などが可能である．しかし，浅原らは，中国語の場合には，tri-gram の規則自体があまり有効ではなく，品詞体系の詳細化が精度の向上に寄与することを実験結果から予測している (浅原, 松本 2002)．

4　YamCha による基本句同定解析

本節では，YamCha を用いた基本句同定解析 (base phrase chunking) 実験について述べる．基本句同定解析とは，形態素解析結果すなわち品詞付与された単語列を入力として，最も下位の構造を同定し，その構造に対して構文的情報を付与する処理である．ここで，最も下位の構造を基本句，基本句に対する構文的情報を句情報と本報告では呼ぶことにする．

このように，基本句同定解析は一段階の構文解析と考えることができる．したがって，構文解析は同定解析を繰り返すことで実現できる (Abney 1991)．工藤らは，SVM に基づく同定解析の段階適用が，日本語の係り受け解析に有効であることを示している (工藤, 松本 2002)．

4.1　学習データ

同定解析の学習データは，形態素解析と同様に CTB を用いた．CTB が表現する構文木では，葉が形態素に相当する．基本句同定解析では，葉に最も近い位置に付与されている構造が基本句であり，形態素が基本句を構成する要素に該当する．すなわち，形態素情報を入力として基本句の区切り位置と句情報を推定する．

4.2　学習文テスト

CTB 全体 (3572 文[9]) を学習データとして無作為に抽出した 1 割の文 (357 文) を解析する学習文テストを行った．この結果を表 9 に示す．学習文テストにおけるテストデータは 7746 の基本句からなる．そのうち YamCha は 7745 の基本句を同定した．このうち 7741 の基本句の同定に成功し，そのうち句情報も正解だったものは 7740 であった．すなわち，学習文テストでは YamCha はほとんどすべての解析に成功した．

表 9: 基本句同定解析の学習文テストの結果

	再現率 (%)	適合率 (%)	F 値 (%)
基本句同定のみ	99.94	99.95	99.94
基本句同定と句情報付与	99.92	99.94	99.93

4.3　未知文テスト

次に，形態素解析実験と同様に，10 分割交差検定による未知文テストを行った．学習の素性は，形態素の文字列情報，品詞情報，基本句の区切りを示す IOB タグとその基本句の句情報である．すべての学習条件は YamCha の標準設定に従っている．

形態素解析実験と同様に再現率と適合率を求めた．未知文テストの結果を表 10 に示す．なお，正解基本句数の平均は 7734.4 であり，YamCha が出力した基本句数の平均は 7691.9 である

表 10: 基本句同定解析の未知文テストの結果

	再現率 (%)	適合率 (%)	F 値 (%)
基本句同定のみ	94.61	95.12	94.86
基本句同定と句情報付与	93.44	93.94	93.69

基本句同定，句情報付与共に比較的高い精度を得られることがわかった．また，句情報付与における誤りのうち出現率の大きい上位 5 種を表 11 に示す．

1 テストセットあたりの学習・解析時間は，表 12 の通りである．なお，形態素解析実験と同一の計算機を使用した場合，モデルファイルのコンパイルに要した時間は約 1 分だった．

基本句同定解析は，形態素解析と比べて高い解析精度を得た．また，学習・解析に要する時間がいずれも短い．これは付与する情報の種類が少ない，およびトークン数が少ないためである．

[9] 形態素解析実験と異なる理由は，見出しなどを対象から除いているためである．

表 11: 誤りパターンの上位 5 種

正解 - 解析	出現率 (%)
IP - VP	47.58 (432/908)
VP - IP	35.35 (321/908)
IP - NP	2.31 (21/908)
UCP - IP	2.20 (20/908)
PRN - IP	1.98 (18/908)

表 12: 基本句同定解析の処理時間

	学習	解析
タグの種類	29 種類	—
文数	約 3200	約 350
トークン数	約 8 万 6 千	約 9500
処理時間	約 2 時間	約 3 分

5　まとめ

本報告では，SVM に基づく言語解析器を用いて，Penn Chinese Treebank を言語資源とした時にどの程度の中国語解析精度が得られるかを報告した．以下にその結果をまとめる．

- SVM に基づく解析器（YamCha）による形態素解析精度は単語単位で約 88% であり，コスト最小法に基づく解析器（MOZ）よりも 4%以上高い．未知語の約 4 割を正しく解析でき，未知語に対する頑健性は高い．ただし計算量に問題があり，解析時間，学習時間共に非常に長い．大量のタグ付きコーパスが入手できる場合は，YamCha，MOZ のいずれを用いても，さらに高精度の解析器（110 万語のコーパスの場合，F 値でそれぞれ約 92%，89%）を実現できる．そのような場合には，解析，学習にかかる時間を考慮すると，MOZ を用いるべきである．なお，中国語形態素解析は日本語のそれと比較して顕著な困難さは見られなかった．
- 基本句同定解析 (base phrase chunking) の精度は約 93%．約 3200 文の学習に約 2 時間，357 文の解析に約 3 分を要する．

謝辞

本研究は，通信・放送機構の研究委託「大規模コーパスベース音声対話翻訳技術の研究開発」により実施したものです．

参考文献

Abney, S. (1991). “Parsing by Chunks.” In Berwick, R. C.,Abney, S. P.,and Tenny, C. (Eds.), *Principle-Based Parsing: Computation and Psycholinguistics*, pp. 257–278. Kluwer Academic Publishers, Boston.

浅原正幸, 松本裕治 (2002). “形態素解析のための拡張統計モデル.” 情報処理学会論文誌, **43** (3), 685–695.

Kreßel, U. H.-G. (1999). “Pairwise Classification and Support Vector Machines.” In Schölkopf, B.,Burges, C. J.,and Smola, A. J. (Eds.), *Advances in Kernel Methods*, pp. 255–268. The MIT Press.

工藤拓, 松本裕治 (2000). “Support Vector Machine を用いた Chunk 同定.” 情報処理学会研究報告 2000-NL-140, pp. 9–16.

工藤拓, 松本裕治 (2002). “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.

前田英作 (2001). “痛快! サポートベクトルマシン–古くて新しいパターン認識手法–.” 情報処理, **42** (7), 676–683.

Sang, E. F. T. K. and Veenstra, J. (1999). “Representing Text Chunks.” In *Proceedings of EACL’99*, pp. 173–179.

平博順, 春野雅彦 (2000). “Support Vector Machine によるテキスト分類における属性選択.” 情報処理学会論文誌, **41** (4), 1113–1123.

Xue, N. and Xia, F. (2000). “The Bracketing Guidelines for the Penn Chinese Treebank (3.0).”. http://www.ldc.upenn.edu/ctb/parseguide.3rd.ch.pdf.

山下達雄, 松本裕治 (2000). “言語に依存しない形態素解析処理の枠組.” 自然言語処理, **7** (3), 39–56.

張玉潔, 山本和英, 坂本仁 (2002). “換言コーパスを利用した中国語換言処理.” 第 8 回年次大会講演論文集, pp. 132–135. 言語処理学会.

付録

A　CTB のタグセット

CTB における全品詞の説明と，全ての句情報の説明を CTB のタグ付与指針 (Xue and Xia 2000) から転記する.

品詞情報：

AD	adverbs	M	measure word (including classifiers)
AS	aspect marker	MSP	some particles
BA	把 in ba-const	NN	common nouns
CC	coordinating conj	NR	proper nouns
CD	cardinal numbers	NT	temporal nouns
CS	subordinating conj	OD	ordinal numbers
DEC	的 for relative-clause etc.	ON	onomatopoeia
DEG	associative 的	P	prepositions (excluding 把 and 被)
DER	得 in V-de const. and V-de-R	PN	pronouns
DEV	地 as the head of DVP	PU	punctuation
DT	determiner	SB	被 in short bei-construction
ETC	tags for 等 and 等等	SP	sentence-final particle
	in coordination phrases	VA	predicative adjective
FW	foreign words	VC	copula 是
IJ	interjection	VE	有 as the main verb
JJ	noun-modifier other than nouns	VV	other verbs
LB	被 in long bei-construction		
LC	localizer		

句情報：

ADJP	adjective phrase	LCP	phrase formed by “XP + LP”
ADVP	adverbial phrase headed	LST	list marker
	by AD (adverb)	NP	noun phrase
CLP	classifier phrase	PP	preposition phrase
CP	clause headed by C (complementizer)	PRN	parenthetical
DNP	phrase formed by “XP + DEG”	QP	quantifier phrase
DP	determiner phrase	UCP	unidentical coordination phrase
DVP	phrase formed by “XP + DEV”	VP	verb phrase
FRAG	fragment		
IP	simple clause headed by I (INFL)		

B　本報告で用いた言語情報資源

以下に，本報告で用いた言語情報資源についてまとめる．

B.1　Penn Chinese Treebank

米国ペンシルバニア大学 (University of Pennsylvania) の Chinese Treebank Project により作成された構文木コーパスである．このプロジェクトは 1998 年夏に始まり，最終版 (LDC2000T48) を 2000 年 12 月に，また同一内容で誤りを修正した Penn Chinese Treebank Version 2.0 (LDC2001T11) を 2001 年に公開した．新華社通信 (新华社通讯, Xinhua News Agency) の 1994 年から 1998 年の 325 記事から構成される，約 10 万語の大きさのコーパスである．

CTB は新华社通讯 の 325 記事に対して，単語分割，品詞付与，構造情報付与されたコーパスである．コーパス作成作業は, (1) 作業者 1 名が全情報を付与する （2） 別の作業者 1 名が点検する，という方法で行った．文字コードには GB コードを使用し，データの書式は English Penn Treebank とほぼ同一である．

入手方法

Linguistic Data Consortium (LDC) より出版されている．そのため，入手方法は通常の LDC のコーパスを入手する方法と同じである．本報告で使用した LDC2000T48 は 2000 年のメンバーに配布可能である．メンバー以外であっても，US$100 にて入手することができる．関連 URI を以下に示す.

Chinese Treebank Project	`http://www.ldc.upenn.edu/ctb/`
LDC	`http://www.ldc.upenn.edu/`
CTB 最終版	`http://www.ldc.upenn.edu/Catalog/LDC2000T48.html`
CTB Version 2.0	`http://www.ldc.upenn.edu/Catalog/LDC2001T11.html`

B.2　人民日報タグ付きコーパス

富士通研究開発中心有限公司 (富士通研究开发中心有限公司) と，北京大学 (北京大学) および人民日報社 (人民日报社) が協力し作成した，中国で最も権威を持ち影響力のある中国全国紙のタグ付きコーパスである．人民日报の 1998 年の新聞記事半年分，約 1,300 万文字＝約 730 万単語からなる．人民日报社新闻信息中心 から，大学や研究所などでの研究利用に限定して，人民元 2,000 元（約 3 万円，実費）にて有償公開している．また，この内 1ヶ月分を無償公開している．

入手方法

正規版の入手に関しては，WWW ページ[10] に記載されている連絡先へ問いあわせる．また，本報告で用いた無償公開版は，北京大学计算语言学研究所の公開ページ[11] にある必要項目を満たすことにより入手できる．

その他，関連 URI を以下に示す．

日本語による説明： http://pr.fujitsu.com/jp/news/2001/08/28.html
北京大学计算语言学研究所：
http://www.icl.pku.edu.cn/Introduction/corpustagging.htm
タグセット一覧： http://www.icl.pku.edu.cn/research/corpus/addition.htm

B.3　京都大学テキストコーパス

京都大学テキストコーパスは，毎日新聞 1995 年 1 月 1 日から 17 日までの全記事，約 2 万文，1 月から 12 月までの社説記事，約 2 万文，計約 4 万文に対して，京都大学の形態素解析システム (JUMAN)，構文解析システム (KNP) で自動解析を行い，その結果を人手で修正したコーパスである．ただし，コーパスとして含んでいるのは形態素・構文の付加情報のみであり，毎日新聞の記事そのものは含まれていない．そのためコーパス本来の形式とするためには別途毎日新聞 CD-ROM が必要である．毎日新聞 CD-ROM を用意し，京都大学テキストコーパスの配布パッケージに含まれるプログラムを使用して完全なコーパスの形式へ変換する．

入手方法

以下のページより入手できる．
http://www-nagao.kuee.kyoto-u.ac.jp/nl-resource/corpus.html

[10] http://www.fujitsu.com.cn/support/
[11] http://www.icl.pku.edu.cn/research/corpus/dwldform1.asp

略歴

吉田 辰巳: 1979 年生．2002 年 豊橋技術科学大学工学部知識情報工学課程卒業．現在，豊橋技術科学大学大学院工学研究科修士課程知識情報工学専攻在学中．自然言語処理，特にテキスト自動要約の研究に従事．

e-mail: gaizi@smlab.tutkie.tut.ac.jp

大竹 清敬: 1973 年生．2001 年豊橋技術科学大学大学院工学研究科博士後期課程電子・情報工学専攻修了．博士（工学）．同年より国際電気通信基礎技術研究所 (ATR) に所属し，現在，音声言語コミュニケーション研究所研究員．自然言語処理，特に換言処理，要約処理，機械翻訳の研究に従事．言語処理学会，人工知能学会，情報処理学会各会員．

e-mail: otake@fw.ipsj.or.jp

山本 和英: 1969 年生．1996 年豊橋技術科学大学大学院工学研究科博士後期課程システム情報工学専攻修了．博士（工学）．同年より国際電気通信基礎技術研究所 (ATR) に所属し，現在音声言語コミュニケーション研究所客員研究員（非常勤）．1998 年中国科学院自動化研究所国外訪問学者．2002 年より長岡技術科学大学電気系講師．換言処理, 要約処理, 機械翻訳, 中国語及び韓国語処理の研究に従事．言語処理学会，情報処理学会，ACL 各会員．

e-mail: yamamoto@fw.ipsj.or.jp

(2002 年 3 月 8 日 受付)
(2002 年 6 月 10 日 再受付)
(2002 年 7 月 24 日 再々受付)
(2002 年 10 月 4 日 採録)