自然言語処理 137－8
音声言語情報処理 31－3
（2000．6．2）

読み正解コーパスのXML化とXSLTの利用

山田篤
京都高度技術研究所/通信総合研究所
〒600-8813 京都市下京区中堂寺南町17 (財) 京都高度技術研究所
E-mail: yamada@astem.or.jp

近年、日本語のテキストコーパスの整備が進み、利用可能な言語資源が蓄積されつつある。本稿では、日本語ディクテーション基本ソフトウェアの開発の一環として行われた大規模テキストコーパスへの読み付与と、形態素解析済の読み正解コーパスの整備について述べる。はじめに、形態素解析システムと読み付与の関係と、既存の形態素解析システムで処理し切れなかった問題について述べる。次に、読み正解コーパスを、特定の形態素解析システムの出力形式に依存しない Extensible Markup Language (XML) で記述することを提案する。XML の採用により、再利用性を向上させるとともに、XSL Transformations (XSLT) などの利用により、コーパス作成作業における各種ツールの共有化が期待できる。

Text Corpus Annotated with Appropriate Pronounciations: an Application of XML and XSLT

Atsushi YAMADA
ASTEM RI/CRL KARC

As a part of the research and development activities for developing Japanese dictation toolkit, we have annotated appropriate pronounciations to text corpus and developed some tools for the annotation. Developing tailered morphological analysis dictionary resolved some problems for constructing appropriate pronounciations annotated corpus, and postprocessing tools the rest. Using Extensible Markup Language (XML), which is independent of any format of Japanese morphological analysis system, we can reuse the annotated corpus in many ways. XSL Transformations (XSLT) helps us to transform XML documents into other forms.

1 はじめに

自然言語処理の研究において、各種コーパスの整備がさかんに行われている。中でも、タグつきコーパスの整備は、コーパスベースのアプローチをとるか否かにかかわらず、重要である。一方で、音声言語研究の観点からは、読みに関する情報が必要となる。例えば、大語彙日本語連続音声認識のための統計的言語モデルの作成においては、正しい読みが付与された言語データを用いる。

本稿では、音声認識のための共通のソフトウェアリポジトリを開発する3ヶ年計画 (1997～1999年度) のプロジェクト[1] [3] の中での言語モデル構築のための各種言語資源・ツールの整備に関する経験をもとにして、大規模テキストコーパスへの読み付与と、読み正解コーパスの整備について述べる。

[1] このプロジェクトは情報処理振興事業協会 (IPA) の「独創的先進的情報技術に係わる研究開発」の支援の下に行われた。

2 形態素解析と読み

はじめに、大規模なテキストコーパスを処理し、正しい読みが付与された大量の言語データを自動的に得るために、形態素解析の結果をもとに、正しい読みを付与することを検討した。即ち、形態素単位で一意に定まる読みについては、形態素辞書に予め正しい読みの情報を与えておくことにより、形態素解析と同時に正しい読みを得ることができると考えた。ところが、1997年当時、このような観点からの読みを与えてくれる形態素解析システムは存在しなかった。調べ得た限りでは、読みを出力

する形態素解析システムは存在するものの、そこで言う読みとはいわゆる現代仮名遣いにもとづく正書法表記であったり、場合によっては実際の読み方とはかけ離れたものであったりした。

そこで、まず

1. 読み表記ガイドラインの設定
2. 読み正解コーパスの整備
3. 形態素解析辞書の整備

を行った。この作業の概要については [4] で報告されているが、既存の形態素解析辞書の読みフィールドの修正は、ある程度は機械的に進めることはできたものの、最終的には人手でチェックする必要があった。なお、形態素解析システムとしては茶筌 [2] を用いた。この時点では、茶筌のプログラム自体に手を加えることは行わなかったため、既存の辞書構造をそのまま用いて、読みフィールドを上書きする形で整備を行った。この結果、昨年度公開したバージョン (IPAdic1.0) では、従来の現代仮名遣いによる読みを出力することはできなくなった。その後、茶筌の辞書構造が変更され、新たに発音フィールドが作られたため、現在では、現代仮名遣いに基づく読みは読みフィールドに、実際の発音に基づく読みは発音フィールドにそれぞれ格納されている。

形態素毎に正しい読みを与えた形態素辞書を用いて形態素解析を行い、その結果得られた読みを、2. で整備した読み正解付コーパスと比較してその正解率を測定したところ、形態素を単位として正解率は約 87%であった。正しく読みが付与できなかったものは以下の 3 点である。

1. **同形異音語**
   形態素としては同じでありながら、複数の読みを持つものが存在する。たとえば、「大勢」は「20 歳以下の若者が大勢を占めた」では「タイセイ」と読まれるのに対し、「大勢の参加者で賑わった」では「オオゼイ」と読まれる。また、同形異音語の中には、「世論」(セロン、ヨロン) のように、読み分けのされ方が明確でなく、どちらの読み方も可能なものも存在する。
2. **数詞**
   連続した数字は位取りして読まれる場合と、一字ずつ読まれる場合がある。たとえば「1300 人」は「センサンビャクニン」と読まれるが、電話番号の一部の「1300」は「イチサンゼロゼロ」と読まれる。さらに次項の音韻変化とも関連し、後続する助数詞によって、数字の読みが変化する場合がある。
3. **音韻変化**
   前後に連なる形態素の影響をうけて、読みが変化する場合がある。たとえば、「袋」は単独では「フクロ」であるが、前に別の名詞がついて「ビニール袋」となると「ブクロ」というように連濁する。助数詞についても、前につく数詞によって読みが変化する。「本」は「1 本」、「2 本」、「3 本」のそれぞれに対して、「ポン」、「ホン」、「ボン」と読まれる。

3 日本語ディクテーションのための読みの整備

既存の形態素解析システムの枠組では処理し切れないものについては、後処理という形で対応することにした。上記のうち、同形異音語については、読み付与という観点からは、読み分けが可能なものには正しい読みのみが付与されることが望ましいが、正解がその中に含まれていれば認識は可能ということから、辞書にすべての読みを併記することによって対応した。

ただし、同形異音語の中には、品詞によって区別できる読みがあることもわかっている。典型的なものとしては、普通名詞として用いられるときと、接辞として用いられるときで読み分けが可能な場合がある。これは、形態素解析システムが正しく品詞を認識するならば、読みも限定することができる。

数詞と助数詞については、先見的な言語的知識を整備し、読み分け規則として構成することとし、この結果は ChaWan というシステムにまとめられた。なお、移植はそれほど困難ではないと考えられるが、ChaWan は茶筌の出力形式に依存している。よって、現状では他の形態素解析システム (JUMAN 等) にそのまま適用することはできない。

活用語まわりのある種の音韻変化は、既存の枠組では処理し切れないという理由から後処理にまわした。具体的には以下の場合である。

1. **一致する活用形**
   「行く」の連用タ接続形と「行う」の連用タ接続形はともに「行った」という表記になる。すると、現在の枠組では、前接する助詞によってある程度の制御はされるものの「行った」が必ずしも正しく解析されるとは限らない。そこで、「行った」に対しては、後処理で、「イッタ」「オコナッタ」の 2 種類の読みを併記してしまうことにする。
2. **語幹の読み変化**
   ア段の音に続くウ音便では [au] が [oo] に変化する。具体的には形容詞・アウオ段のうち、ア段の「長い」が「長う」になった場合、語幹の読みは「ナガ」から「ナゴ」に変化する。同様に、動詞・五段ワ行の

「疑う」は連用タ接続形において「ウタガウ」から「ウタゴー」に読みが変化する。しかし、現在の枠組ではカ行変格活用などの特殊な場合を除いては、語幹の読み変化には対応していないので、このような読みの変化は後処理で対応する。なお、このとき、仮名表記の文がうまく解析できないという問題が残る。このため、「うたがうた」は解析できてしまうが「うたごうた」は解析できない。

4 読み正解コーパスのXML化

上で述べたChaWanや後処理ツールは、いずれも茶筌の特定の出力形式に依存している。そこで、これらのツールを用いるための一つの方法は、処理対象のフォーマットを何らかの方法で変換して、茶筌の出力形式にあわせることである。この出力形式は、デフォルトでは1行1形態素のタブ区切りである。しかしながら、タブ区切りのそれぞれの列に何が格納されるのかは別途記録しておかなければならない。大体において原表記は第1カラムに書かれることが多いが、特にそのような取り決めがあるわけでもない。また、読みと発音の区別などは、実際に内容を見てみないと判断できない。さらに、複数のバージョンの辞書を使い分けて、比較するような場合には、当該データがどの辞書を用いて生成されたものかは、別途記録しておかねばならない。

これらの問題は、それぞれの記述要素の意味が明らかな共通のフォーマットを採用するとともに、解析システムや用いた辞書などの情報をメタデータとして、解析データそのものと一緒に格納することにより解決できる。

このような目的に合致するものとして、近年Extensible Markup Language (XML) [2]が提唱され、自然言語処理の分野でもGDA (Global Document Annotation)などで用いられている。XMLでは要素は木構造をなし、各要素はそのタイプ毎に属性とその値のペアを持つことができる。用いた形態素解析システムや辞書のバージョン、人手修正をした場合のアノテータの情報等はメタデータとして、いずれかの要素の属性に格納できる。

XMLの構造設計では、要素と属性の区別が必要になるが、単純にコーパスは文の並び、文は形態素の並び、と考え、コーパス、文、形態素を要素として、形態素のコンテンツに原テキストの文字列を持たせることとする。形態素解析結果の情報はすべて各要素毎の属性として格納する。このようにして設計したDocument Type Definition (DTD)は図1のようになる。

```
<!ELEMENT corpus (sen+)>
<!ATTLIST corpus
    analyzer CDATA #IMPLIED>
<!ELEMENT sen (morph+)>
<!ELEMENT morph (#PCDATA)>
<!ATTLIST morph
    yomi CDATA #IMPLIED
    pronounciation CDATA #IMPLIED
    base_form CDATA #IMPLIED
    part_of_speech CDATA #IMPLIED
    inflection_type CDATA #IMPLIED
    inflected_form CDATA #IMPLIED>
```

図1: 読み正解コーパスのためのDTD

たとえば、ChaSen (Ver.2.03b1)[3]、JUMAN (Ver.3.61)[4]、sumomo (Ver.1.3)[5]のそれぞれの出力は図2のようになるが、これを上記のXML形式に変換すると、図3のようになる。

```
カネ    カネ    カネ    名詞-一般
の      ノ      の      助詞-連体化
力      チカラ  力      名詞-一般
も      モ      も      助詞-係助詞
十分    ジュウブン      十分    名詞-形容動詞語幹
知っ    シッ    知る    動詞-自立       五段・ラ行
て       テ      て      助詞-接続助詞
い      イ      いる    動詞-非自立     一段    連用形
た      タ      た      助動詞  特殊・タ        基本形
。      。      。      記号-句点
EOS
```

(a) ChaSen

```
カネ        (かね)      カネ        人名
の          (の)        の          接続助詞
力          (ちから)    力          普通名詞
も          (も)        も          副助詞
十          (じゅう)    十          数詞
分          (ふん)      分          名詞性名詞助数詞
知っていた  (しっていた)  知っている  動詞        母音動詞  タ形
。          (。)        。          句点
EOS
```

(b) JUMAN

```
                                "文頭"  "文頭"  "文頭"
カネ    かね    カネ    "普通名詞"      "普通名詞"      "普通名詞"
の      の      の      "助詞"  "体言後接助詞"  "格助詞"
力      ちから  力      "普通名詞"      "普通名詞"      "普通名詞"
も      も      も      "助詞"  "格助詞後接助詞"        "係助詞"
十      じゅう  十      "数詞"  "数詞"  "数詞"
分      ふん    分      "単位"  "単位"  "単位"
知って  しって  知っている  "動詞"  "動詞"  "一段動詞語幹"
た      た      た      "助動詞"        "動詞連用形・形容詞連用形かっ・形容動詞だっ後接助詞 or 助動詞た"      "形容詞終止・連体形・助動詞終止・連体形た・ぬ"
。      。      。      "記号"  "句点"  "句点"
```

(c) Sumomo

図2: 形態素解析システム毎の出力

なお、これらのXMLインスタンスは各形態素解析システムの出力を読み込んで直接変換するscriptによって

[2] http://www.w3c.org/TR/REC-xml/

[3] http://cl.aist-nara.ac.jp/lab/nlt/chasen/

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

[5] http://www.t.onlab.ntt.co.jp/sumomo/index.html

```
<?xml version="1.0" encoding="EUC-JP"?>
<corpus analyzer="ChaSen_2.03b1">
<sen>
<morph yomi="カネ" base_form="カネ" part_of_speech="名詞-一般" inflection_type="" inflected_form="">カネ</morph>
<morph yomi="ノ" base_form="の" part_of_speech="助詞-連体化" inflection_type="" inflected_form="">の</morph>
<morph yomi="チカラ" base_form="力" part_of_speech="名詞-一般" inflection_type="" inflected_form="">力</morph>
<morph yomi="モ" base_form="も" part_of_speech="助詞-係助詞" inflection_type="" inflected_form="">も</morph>
<morph yomi="ジュウブン" base_form="十分" part_of_speech="名詞-形容動詞語幹" inflection_type="" inflected_form="">
十分</morph>
<morph yomi="シッ" base_form="知る" part_of_speech="動詞-自立" inflection_type="五段・ラ行" inflected_form="連用タ接
続">知っ</morph>
<morph yomi="テ" base_form="て" part_of_speech="助詞-接続助詞" inflection_type="" inflected_form="">て</morph>
<morph yomi="イ" base_form="いる" part_of_speech="動詞-非自立" inflection_type="一段" inflected_form="連用形">い
</morph>
<morph yomi="タ" base_form="た" part_of_speech="助動詞" inflection_type="特殊・タ" inflected_form="基本形">た
</morph>
<morph yomi="。" base_form="。" part_of_speech="記号-句点" inflection_type="" inflected_form="">。</morph>
</sen>
</corpus>
```

(a) ChaSen

```
<?xml version="1.0" encoding="EUC-JP"?>
<corpus analyzer="JUMAN_3.61">
<sen>
<morph yomi="かね" base_form="カネ" part_of_speech="名詞-人名" inflection_type="*" inflected_form="*">カネ</morph>
<morph yomi="の" base_form="の" part_of_speech="助詞-接続助詞" inflection_type="*" inflected_form="*">の</morph>
<morph yomi="ち か ら" base_form="力" part_of_speech="名 詞-普 通 名 詞" inflection_type="*" inflected_form="*">力
</morph>
<morph yomi="も" base_form="も" part_of_speech="助詞-副助詞" inflection_type="*" inflected_form="*">も</morph>
<morph yomi="じゅう" base_form="十" part_of_speech="名詞-数詞" inflection_type="*" inflected_form="*">十</morph>
<morph yomi="ふん" base_form="分" part_of_speech="接尾辞-名詞性名詞助数辞" inflection_type="*" inflected_form="*">
分</morph>
<morph yomi="しっていた" base_form="知っている" part_of_speech="動詞-*" inflection_type="母音動詞" inflected_form="
タ形">知っていた</morph>
<morph yomi="。" base_form="。" part_of_speech="特殊-句点" inflection_type="*" inflected_form="*">。</morph>
</sen>
</corpus>
```

(b) JUMAN

```
<?xml version="1.0" encoding="EUC-JP"?>
<corpus analyzer="sumomo_1.3">
<sen>
<morph yomi="かね" base_form="カネ" part_of_speech="普通名詞" inflection_type="普通名詞" inflected_form="普通名詞">
カネ</morph>
<morph yomi="の" base_form="の" part_of_speech="助詞" inflection_type="体言後接助詞" inflected_form="格助詞の">の
</morph>
<morph yomi="ちから" base_form="力" part_of_speech="普通名詞" inflection_type="普通名詞" inflected_form="普通名詞">
力</morph>
<morph yomi="も" base_form="も" part_of_speech="助詞" inflection_type="格助詞後接助詞" inflected_form="係助詞">も
</morph>
<morph yomi="じゅう" base_form="十" part_of_speech="数詞" inflection_type="数詞" inflected_form="数詞">十</morph>
<morph yomi="ふん" base_form="分" part_of_speech="単位" inflection_type="単位" inflected_form="単位">分</morph>
<morph yomi="しってい" base_form="知っている" part_of_speech="動詞" inflection_type="動詞" inflected_form="一段動詞
語幹">知ってい</morph>
<morph yomi="た" base_form="た" part_of_speech="助動詞" inflection_type="動詞連用形・ 形容詞連用形かっ・ 形容動詞だっ
後接助詞 or 助動詞た" inflected_form="形容詞終止・連体形・ 助動詞終止・ 連体形た・ぬ">た</morph>
<morph yomi="。" base_form="。" part_of_speech="記号" inflection_type="句点" inflected_form="句点">。</morph>
</sen>
</corpus>
```

(c) Sumomo

図 3: 形態素解析システム毎の XML 文書

生成した。

この DTD はあくまでも一例であるが、XML 化することにより、形態素解析システムの出力で暗黙のうちに仮定されていた構造を XML 文書の中で明示的に表すとともに、メタデータを格納できている。また、このような形式にしておくことにより、その後の読み正解コーパスの整備作業においても、各形態素解析システム毎に特化するのではなく、汎用のツールが作成できるメリットがある。ただし、採用している文法体系の違いによって、変換等には個別の規則が必要になるかもしれない。

また、この DTD では要素のコンテンツとしては、原テキストの文字列のみを含んでいるため、これを通常の HTML ブラウザで見ると、図 4 に示すようになる。

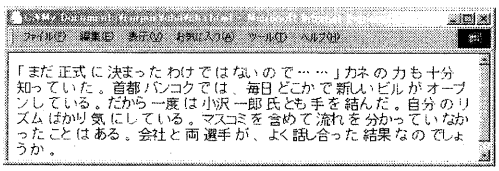

図 4: HTML ブラウザによる表示

一方、Internet Explorer 5[6] ではXML文書を読み込んで木構造で表示することもできる。あるいは、Apacheプロジェクトで作成されているXMLパーサ Xerces[7] に付属のTreeViewerを用いても、木構造の表示が可能である (図5)。

また、Xeena[8] はIBMによるフリーのXMLエディタである。これにXML文書を読み込ませた様子を図6に示す。

5 XSLTによる変換

XSL Transformations (XSL)[9] は、本来 Extensible Stylesheet Language (XSL)[10] の一部として主に、XML文書どうしの変換を目的として作られた言語である。XSLTはいまだW3C勧告となっていないが、XSLTは一足先に勧告となった。XSLTの処理系としては例えば、Xalan[11] などがある。

XSLTを用いることで、先のXML文書を容易に他の形式に変換することができる。例えば、先の木構造表示は必ずしも見やすいものではなかった。そこで、表示のためにXML文書をHTML化することによって、ルビ表示 (図7) やテーブル状の表示 (図8) を行うことができる。

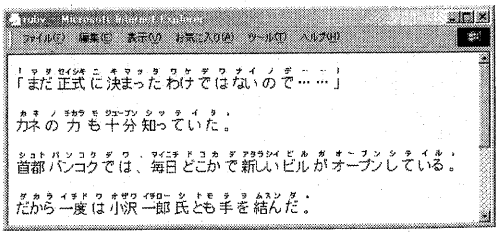

図 7: 読み正解コーパスのルビ表示

さらに、読みの部分をテキストボックスに変換したり、複数読み候補をHTMLフォームのセレクトボックスに変換すれば、アノテータがWebインタフェースを用いて作業できる環境を構築することも容易である。また、先に述べた後処理プログラムによる読み修正も、このXML文書に対する変換の一環として行うことも考えられる。

これらの変換において、XML文書の部分を指定するための言語である XML Path Language (XPath)[12] を

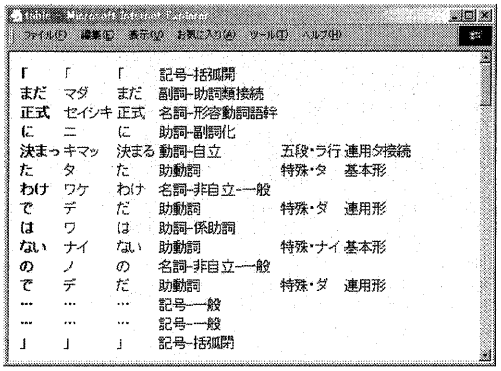

図 8: 読み正解コーパスのテーブル状表示

用いるが、これによって、構造に基づく検索が可能になる。例えば特定の形態素 (複数でもよい) を含んだ文を取り出すといったことが容易に行える。

もちろん、XSLTを用いずに、直接XML文書を操作するプログラムを作成してもよい。その場合でも、XML文書を取り扱うための各種ツール、あるいはSAX (Simple API for XML)[13] やDocument Object Model (DOM)[14] ベースのパーサを利用することができる。たとえば、読み正解コーパスの整備過程では、正解コーパスに付与された読みを、別の形態素解析システムの出力に移行する作業が必要となる。区切りが一致しない場合に、DPを用いて移行するなどの工夫が必要となるが、データ構造を統一することにより、形態素解析システムのフォーマット毎に移行プログラムを作成する必要はなくなる。

6 おわりに

本稿では、日本語ディクテーション基本ソフトウェアの開発の一環として行われた大規模テキストコーパスへの読み付与と、形態素解析済の読み正解コーパスの整備について述べた。はじめに、形態素解析システムによる読み付与と、後処理にまわされた問題について述べた。次に、読み正解コーパスを、特定の形態素解析システムの出力形式に依存しない Extensible Markup Language (XML) で記述することを提案した。XMLの採用により、再利用性を向上させるとともに、XSL Transformations (XSLT) などの利用により、コーパス作成作業における各種ツールの共有化が期待できる。今後は、形態素解析済読み正解コーパスの再利用性を高めるとともに、語構成や複合語の表現等の表現能力を追加する予定である。

[6] http://www.microsoft.com/ie/ie5/
[7] http://xml.apache.org/xerces-j/index.html
[8] http://www.alphaworks.ibm.com/tech/xeena/
[9] http://www.w3c.org/TR/xslt/
[10] http://www.w3c.org/Style/XSL/
[11] http://xml.apache.org/xalan/index.html
[12] http://www.w3c.org/TR/xpath/
[13] http://www.megginson.com/SAX/index.html
[14] http://www.w3c.org/MarkUp/DOM/

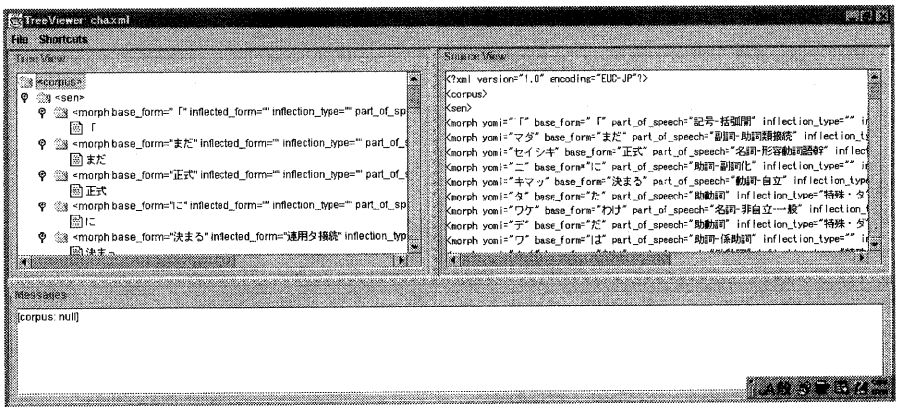

図 5: TreeViewer による表示

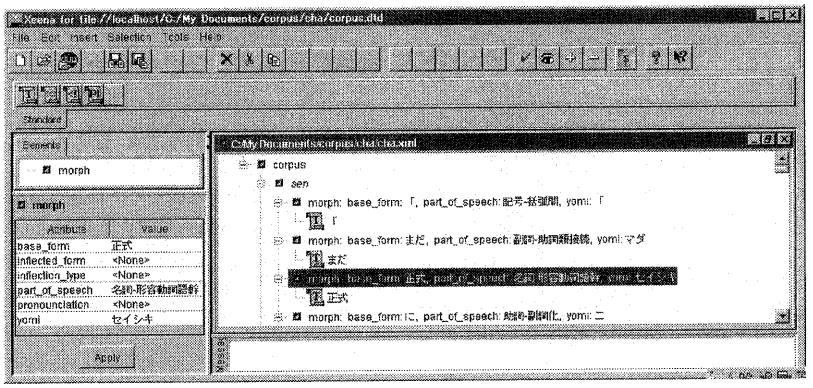

図 6: Xeena を用いた編集

謝辞: 情報処理振興事業協会 (IPA) の独創的先進的情報技術に係わる研究開発「日本語ディクテーション基本ソフトウェア」開発プロジェクトの関係各位、特に言語モデルサブグループの諸氏に感謝します。また、通信総合研究所関西支所知的機能研究室の各位、奈良先端科学技術大学院大学「茶筌」開発グループの各位に感謝します。さらに、言語・音声資源として毎日新聞社による CD-毎日新聞データ集、また技術研究組合新情報処理開発機構による RWC テキストデータベース、音響学会データベース委員会による JNAS 新聞記事読み上げコーパスを利用しています。これらを提供し、また利用を許可して下さり、ご協力頂いた関係各位に感謝します。

参考文献

[1] NHK 放送文化研究所: NHK 日本語発音アクセント辞典 新版 (1998).

[2] 松本, 北内, 山下, 平野, 松田, 浅原：“日本語形態素解析システム『茶筌』version 2.0 使用説明書 第二版”, Information Science Technical Report NAIST-IS-TR99012, 奈良先端科学技術大学院大学 (1999).

[3] 河原, 李, 小林, 武田, 峯松, 伊藤, 山本, 山田, 宇津呂, 鹿野：“日本語ディクテーション基本ソフトウエア (98 年度版) の性能評価”, 情報処理学会 音声言語情報処理研究会, 99-SLP-26-6, (1999).

[4] 伊藤, 山田, 天白, 山本, 踊堂, 宇津呂, 山本：“日本語ディクテーションのための言語資源・ツールの整備”, 情報処理学会 音声言語情報処理研究会, 99-SLP-26-5, (1999).