コーパスシステム『Co-Chu』の開発
—MeCab 拡張データ処理機能について—

ラニガン　マシュー（中部大学大学院国際人間学研究科）†

Development of the Corpus System “Co-Chu”
—Regarding the MeCab Data Processing Extensions—

Matthew Lanigan (Chubu University)

要旨

本発表では、音声データを書き起こしたものを形態素解析にかける際に起こる問題点とその解決方法の一つとして、MeCab 拡張データ処理システムについて報告する。コーパスシステム『Co-Chu』は、コーパス検索だけでなく、コーパス開発のツールとして開発された。『名大会話コーパス』『日本語学習者会話データベース』『BTSJ による日本語話し言葉コーパス』をシステムに入れたところ、音声書き起こしコーパスに現れる学習者の誤用、言いよどみやフィラーなど、形態素解析のエラーを及ぼすものが様々あった。それらを排除する手段もあるが、そうすると分析対象とならないため、それらの問題点を補うシステムが必要となる。そこで、『Co-Chu』の開発の際に、特定の読みや出現形を選定するタグや辞書エントリーを一時的に導入するタグを付け、MeCab 拡張タグを開発した。本発表では、『Co-Chu』の MeCab 拡張データ処理システムとその仕組みについて報告する。

１．はじめに

話しことばコーパスの開発が困難になる原因がいくつか考えられる。

まず第 1 に、形態素解析を行う際、データがきれいでなければ、エラー率が非常に高まる可能性があり、話し言葉には様々な「きれいでない」要素が含まれている。例えば、言いよどみやフィラーなどが問題点として指摘されている(内元、野畑、山田、他 2003)。

第 2 に、コーパス開発および分析のためのツールは色々あるが、コンピューター技術に関する知識があまりなければ、使いこなすのは難しいと言えるだろう。

第 3 に、様々なツールがあっても、自分のデータで利用できる、『中納言』や『NINJAL-LWP for BCCWJ』のような強力なツールは少ない。

そこで、オープンソースソフトウェア（OSS）のコーパスシステム『Co-Chu』の開発を試みた。本発表では、『Co-Chu』の MeCab 拡張データ処理機能を中心に報告する。

２．『Co-Chu』の概要

コーパスシステムというのは、大きく分けて、コーパス開発とコーパス分析という２面で構成される。『Co-Chu』とは「中部大学コーパスシステム（Chubu University Corpus System）」の略である。現在システムは開発中であり、公開できるものになっていないが、近日中には公開予定である。

概要に入る前、本システムは『BCCWJ』や『日本語話し言葉コーパス』のような大規模コーパスの開発に利用されるためには作られていないことを注意しておきたい。『Co-

† lanigan@isc.chubu.ac.jp

Chu』に含まれている MeCab 拡張タグなどのコーパス開発ツールはほぼ必ず手作業を必要とするものであり、データの量が多ければ多いほど手におえなくなるだろう。コンピューター技術に詳しくない個人の言語研究者や小グループで開発されているコーパスを念頭に開発しようとしている。しかし、大規模コーパスのために作られていないとはいえ、データの量が非常に多くても機能するように配慮した。

２.１　システムの構造

『Co-Chu』はデータベース、アプリケーション・プログラミング・インターフェース（API）、ユーザー・インターフェース（UI）の 3 階層構造になっている。この構造を使用することにより、システムの拡張が容易になると期待できる(日本 OSS 推進フォーラム 2015)。

２.１.１　データベース

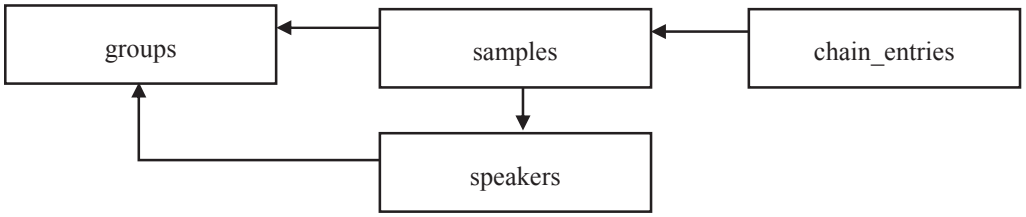

図 1 データベース構造の概要

システムの基盤となるのはデータベースである。『Co-Chu』のデータベースは PostgreSQL という OSS のリレーショナルデータベース（RDB）を利用している。他にもあるが、最も重要なテーブル（groups, samples, speakers, chain_entries の 4 つ）とその関係を図 1 に示している。

表 1 テーブルの構造

groups	
id	INTEGER
parent_id	INTEGER
name	TEXT
metadata	JSONB

samples	
id	INTEGER
group_id	INTEGER
speaker_id	INTEGER
name	TEXT
metadata	JSONB

speakers	
id	INTEGER
group_id	INTEGER
metadata	JSONB

スピーカーのテーブルがあるが、話し言葉に限られていないことを注意しておきたい。形態素解析などにおいて、話し言葉データの処理が特に困難であるため、本システムの大半の機能は話し言葉の処理のために向けられている。

サンプルというのは基本的に発話（、あるいは書き言葉の場合の文章）を示し、グループはサンプルの集まり、つまりサブコーパスにあたるものとする。BCCWJ において「サンプル」というのは作品や記事などを表すが、本システムにおいて作品や記事はグループになるのである。次に、スピーカーは発話者の関係を表し、基本的に話し言葉データでし

か利用されない。

表 1 で見られるように、以上の 3 つのテーブルは全てメタデータコラムがある。これは JSON フォーマットの非構造化データであり、PostgreSQL 9.4 の JSONB タイプによってインデックスされている。

最後に、本システムのコアとなるテーブルは chain_entries である。このテーブルには、一つのサンプルの形態素解析結果をチェインとして保存してある。つまり、MeCab による形態素解析結果に加え、表 2 に見られるように、ID 番号（id）と親 ID 番号（parent_id）があり、形態素の連鎖になる。

表 2「chain_entries」テーブルのデータ例（一部のコラム）

sample_id	id	parent_id	surface
89067	3987301	(NULL)	これ
89067	3987302	3987301	は
89067	3987303	3987302	コーパス
89067	3987304	3987303	システム
89067	3987305	3987304	で
89067	3987306	3987305	ある
89067	3987307	3987306	。
89067	3987308	3987307	(BOS/EOS)

２.１.２　API

第 2 階層になるのは Ruby で作られている REST API である。このインターフェースを通して、UI がデータベースにアクセスする。また、API には MeCab とのインターフェースがあり、そこに MeCab 拡張データ処理機能が入る。つまり、API が UI からデータ処理リクエストを受け、MeCab にデータを転送する前に拡張タグをプロセスしておく。それから、MeCab から形態素解析の結果を読み、タグに含められた指示に従い、データを処理し、データベースへ転送する。具体的なタグについては後述する。

２.１.３　UI

『Co-Chu』の第 3 階層である UI を入れ替えることができるが、グループの間での共有を考えてウェブ・インターフェースにした。未公開データなどは、セキュリティーが重要であると考えられ、ユーザー登録を必要とする。現在管理人しか新しいアカウントが作成できないが、オープンな設定にすることも考慮している。

２.２　コーパス開発機能

API／UI には様々なコーパス開発の際に役立つと思われる機能を付加しており、ここでは、開発済みもしくは開発予定の機能を簡単に紹介する。

- 組み入れ無制限のグループ構成
- WYSIWYG データ移入ツール（TXT、CSV/TSV、Excel）
- MeCab の n-best 機能に基づいたビジュアル・エラー処理ツール
- グループ特定のユーザー辞書

MeCab 拡張タグについては後述する。

２.３　コーパス分析機能

次に　簡単にコーパス分析機能を紹介する。

- 形態素連鎖の詳細検索
- いくつかのアウトプット形式（KWIC など）
- TSV ファイルへの輸出
- N-gram に基づいたコロケーション(Wei and Li 2013)
- 検索結果統計とグループ別比較

３．MeCab 拡張データ処理機能

上述したように、MeCab 拡張データ処理機能は API の一部であり、API が MeCab とインターフェースする前後に行われている。執筆時点で作成されているタグは以下の 5 種類である。フォーマットは基本的に{{<command>{<options>}}}の形をとっている。

command というは短いローマ字のタグセレクターであり、options はタグそれぞれで異なるパラメーターである。しかし、一般的にタグの最初のパラメーターはターゲットとなっている語である。

３.１　読み選定タグ {y{A|B}}

読み選定タグ（y）によって、ある文字の特定読みを選択するタグである。パラメーターは対象語と読みの 2 つである。例えば、{y{昨日|さくじつ}}で見られるように、「昨日」の中の「さくじつ」の読みを選択している。

この機能が重要なのは、書き言葉と異なり、話し言葉はもともと「字」ではなく「音」であるため、話し言葉コーパスの開発には発音が最も重要な要素である。それにもかかわらず、単に「昨日」を形態素解析に入れると、ほぼ確実に「きのう」の読みが出力される。このような例が他にも様々ある。例えば、「後」「明日」「家」などが挙げられる。

最後の「家」については、話し言葉でたまに「俺(おれ)ん家(ち)」のような例がみられるが、MeCab と UniDic で形態素解析を試みると、

出現形	出現形 発音形	語彙素 発音形	語彙素	品詞
俺	オレ	オレ	俺	代名詞
ん	ン	ノ	の	助詞-格助詞
家	イエ	イエ	家	名詞-普通名詞-一般

という結果があり、「家(いえ)」として解析されている。また、「俺んち」の形にしておいても、

出現形	出現形 発音形	語彙素 発音形	語彙素	品詞
俺	オレ	オレ	俺	代名詞
ん	ン	ンー	んー	感動詞-フィラー
ち	チ	チ	チ	記号-一般

のように出力され、「ち」が記号となっている。MeCab の n-best 機能を利用すれば、正しい読みを選択することができ、発音と合致した結果にできる。

以上の例は UniDic によるエラーであるとしても、発音を重視しながら形態素解析を行う際に必ず他の語にも現れる問題である。

３.２　語形選定タグ{w{A|B}}

読み選定タグと違い、語形選定タグはただ選択するのではなく、結果的にある語形に新しい出現形を作り上げるタグである。最初のパラメーターは読みタグと同様、対象語である。しかし、このタグの第 2 のパラメーターは語形となっている。なぜなら、日本語学習者の誤用などを表す使用の仕方が考えられる。例えば、{w{きー|来}}た では、ある学習者が発音を間違え、「来た」の「き」を長音にする。本来ならこれは「来た」に処理されるか、形態素解析後の手作業で直されるか、エラーになるかだが、このタグを利用し、「来」の新しい出現形「きー」を特定な箇所に限ってつけることができる。

つまり、このタグによって、意味も発音も保存され、以上の例からの出力は以下のようになる。

出現形	出現形 発音形	語彙素 発音形	語彙素	品詞	活用型	活用形
きー	キー	クル	来る	動詞-非自立可能	カ行変格	連用形-一般
た	タ	タ	た	助動詞	助動詞-タ	終止形-一般

API にはこのタグは MeCab にデータを転送する前に、A を B に置き換え、結果の B に相当する語の出現形を A に置き換える。

３.２.１　誤用 {err{A}}

このタグは語形制定タグに関連するショートカットであり、他のタグと違い、A は語だけではなく、他のタグを入れることができる。このタグによって、「誤用」というエントリーが対象語の用法コラムに追加される。例えば、語形選定タグの例につけることが、{err{{w{きー|来}}}}のようにできる。

３.３　辞書エントリータグ {d{A|B1,B2,…,Bn}}

辞書エントリータグによって一時的に辞書エントリーを追加することができる。第 2 のパラメーター（B1,B2,…Bn）は UniDic のユーザー辞書のフォーマットになる。基本的にこのタグを直接使う場が少なく、他のタグが利用するためにある。

３.３.１　フィラー {f{A}}

フィラータグによって、何かを 1 語のフィラーとして扱わせることができる。例えば、状況により「ん」が助詞の「の」として認識される場合があり、それを{f{ん}}にすればフィラーになる。

API には、このタグをプロセスするとき、対象語のみがフィラーに変えられるために、まず対象語にプレースホルダ―を置き換える。プレースホルダ―のエントリーを一時的に

ユーザー辞書に追加し、形態素解析を行う。それから、プレースホルダ―が結果に出たら、また対象語をそこに置き換える。

４．まとめ

以上『Co-Chu』の MeCab 拡張データ処理機能を中心に報告した。本システムはコンピューターに詳しくない研究者などが同じインターフェースを通してコーパス開発と分析ができる。また、話し言葉の形態素解析とデータ処理に役立つシステムである。現在、読み選定タグなど、MeCab のユーザー辞書と n-best 機能に基づいたいくつかの MeCab 拡張タグを利用することができる。システムのタグをさらに増やし、話し言葉データの本発表に触れていないの問題点（同時発話や相杏など）に対する解決策は今後の課題としたい。

謝辞

本研究を進めるにあたり、『Co-Chu』のテスター役を含め実際にシステムをご利用くださっている中部大学の山本裕子先生、本間妙先生の貴重なご助言に厚く御礼申し上げます。また多岐にわたるご指導を賜りました小森早江子先生に、心より感謝申し上げます。

参考文献

内元清貴、野畑周、山田篤、関根聡、井佐原均（2003）「日本語話し言葉コーパスの形態素解析」『言語処理学会　第 9 回年次大会　発表論文集』pp.113‐116.

日本 OSS 推進フォーラム（2015）「Ⅱ-3-5. OSS による Web3 層アプリケーション」2015 年 2 月 3 日参照. http://ossforum.jp/en/node/813.

Wei, Naixing, and Jingjie Li (2013). “A New Computing Method for Extracting Contiguous Phraseological Sequences from Academic Text Corpora.” *International Journal of Corpus Linguistics*. 18:4, pp.506–35. doi:10.1075/ijcl.18.4.03wei.

関連 URL

『Co-Chu』http://co-chu.org/. 執筆時点で未完成.

『名大会話コーパス』『日本語学習者会話データベース』2014 年 12 月参照. http://dbms.ninjal.ac.jp/nknet/

『BTSJ による日本語話し言葉コーパス』2015 年 2 月 3 日参照. http://www.tufs.ac.jp/ts/personal/usamiken/btsj_corpus.htm

『MeCab』2015 年 2 月 3 日参照. http://mecab.googlecode.com/svn/trunk/mecab/doc/index.html

『UniDic』2015 年 2 月 3 日参照. http://www.ninjal.ac.jp/corpus_center/unidic/

『中納言』2015 年 2 月 3 日参照. https://chunagon.ninjal.ac.jp/

『NINJAL-LWP for BCCWJ』2015 年 2 月 3 日参照. http://nlb.ninjal.ac.jp/

『PostgreSQL』2015 年 2 月 3 日参照. http://www.postgresql.org/

『Ruby』2015 年 2 月 3 日参照. https://www.ruby-lang.org/

『AngularJS』2015 年 2 月 3 日参照. https://angularjs.org/