学術論文執筆のための仮名漢字変換システム

高橋 文彦* 前田 浩邦† 森 信介‡

* 京都大学大学院情報学研究科
† サイボウズ株式会社
‡ 京都大学学術情報メディアセンター

1 はじめに

自然言語処理の一つの目的は，人の言語活動を支援することである．様々な研究の成果をこの目的に応用することが可能であると考えられる．一方で，自然言語処理の研究者自身が研究の応用を実際に使っている例は多くはない．簡単に思いつく例として，論文執筆時に自動誤り訂正を用いて論文の校正を行うことや，自動要約技術を用いて抄録や発表資料を自動生成すること，ある研究論文を別の言語で発表する際に機械翻訳を用いることなどが挙げられよう．実際に使うと，論文の評価基準とユーザーの評価との違いに代表される様々な問題を体感することができるであろう．

このような背景に鑑み，我々は自然言語処理の論文を学習コーパスとして活用する仮名漢字変換システムを作成した．本システムは，統計的仮名漢字変換 [1] に生コーパスの部分文字列を変換候補として列挙する拡張 [2] に基づくサーバ．及び，テキストエディタ Emacs 上で動作する仮名漢字変換クライアントからなる．したがって，本システムは LaTeX 等の論文執筆等に使用でき，過去の自然言語処理の論文に含まれる専門用語等が変換候補に挙がることで論文執筆が容易になると期待できる．

本論文では，サーバとクライアントについて述べ，仮名漢字変換の変換結果について述べる．また，今後様々な学術分野に対応することを考えて，別の分野のテキストを学習コーパスに入れることの影響についても実験的に考察する．

2 仮名漢字変換サーバ

本章では仮名漢字変換サーバについて説明する．自然言語処理の論文にはアノテーション情報がない．そこで，アノテーションされていないコーパスに対し確率的に単語分割位置を付与し，専門用語などといった単語を変換候補として列挙することに利用する．また，仮名漢字変換サーバとして単語と入力記号列の組を単位とする確率的モデルによる仮名漢字変換 [3] を用いた．

2.1 擬似確率的タグ付与コーパス

専門用語などといった単語を仮名漢字変換の変換候補として列挙するために，擬似確率的タグ付与コーパスを作成する．擬似確率的タグ付与により，一般的な自動タグ付与ではアノテーション出来ない未知語候補がコーパスにアノテーションされる．次の例で説明する．

擬似確率的タグ付与コーパスの例

ガ/が | 格/かく | の/の | 用法/ようほう
例文/れいぶん | の/の | ガ格/がかく

1行目の文では「ガ格」を「ガ」と「格」に分割しているが，2行目の文では「ガ格」を1単語としてコーパスにアノテーションされている．これは「ガ」と「格」の間が確率的に分割され，単語境界有無の揺れが生じた結果である．この例では,「ガ」と「ガ格」という未知語候補がコーパスにアノテーションされているので，仮名漢字変換利用者が「がかく」と入力すれば「ガ格」に変換できる．

本研究では擬似確率的タグ付与コーパスを作成するために，アノテーション情報のないテキストから文献 [3] の方法を用いて単語境界・読み情報を付与する．擬似確率的タグ付与コーパスは，確率的タグ付与コーパスの高コストな計算量を軽減する方法として，タグ付与済みコーパスで確率的タグ付与コーパスを近似する方法を用いている．具体的には，まずコーパスに対して以下の処理を最初の文字から最後の文字まで $(1 \leq i \leq n_r)$ 行なう．

1. 文字 x_i を出力する．

*takahasi@ar.media.kyoto-u.ac.jp
†hirokuni.maeta@gmail.com
‡forest@i.kyoto-u.ac.jp

表 1: 実験で使用するコーパス

学習	分野	文数	単語数	文字数
BCCWJ-train	一般分野	56,753	1,324,951	1,911,660
BCCWJ-nonc	一般分野	716,154	16,749,959	23,782,812
NLP-train	自然言語処理	43,173	1,552,650	2,504,356
MMH-train	医学	50,915	1,561,245	2,141,620
テスト	分野	文数	単語数	文字数
BCCWJ-test	一般分野	6,025	148,929	212,261
NLP-test	自然言語処理	265	29,368	41,738
MMH-test	医学	1,000	8,666	12,775

2. 0 以上 1 未満の乱数 r_i を発生させ P_i と比較する．$r_i < P_i$ の場合には単語境界記号を出力し，そうでない場合には何も出力しない．

これにより，確率的単語分割コーパスに近い単語分割済みコーパスを得ることができる．これを擬似確率的単語分割コーパスと呼ぶ．同様にして，擬似確率的単語分割コーパスの各単語に対して，最初の単語から最後の単語までその都度発生させた乱数と読みの確率の比較結果から該当単語の読みを決定する．これにより，確率的読み付与コーパスに近い読み付与済みコーパスを得ることができる．これを擬似確率的単語分割読み付与コーパスまたは，単に擬似確率的タグ付与コーパスと呼ぶ．単語境界確率と読み付与確率は，点予測を用いて，単語分割読み付与済みコーパスから推定したロジスティック回帰に基づくモデルで計算する．ツールとして KyTea[4] を用いた．

2.2 表記と読みの組を単位とする言語モデル

仮名漢字変換サーバとして，文献 [3] の単語と読みの組を単位とする言語モデルを用いる．確率的モデルによる仮名漢字変換 [2] は，キーボードから直接入力可能な入力記号 $\mathcal{Y}$ の正閉包 $\boldsymbol{y} \in \mathcal{Y}^+$ を入力として，日本語の語彙の正閉包 $\boldsymbol{w} \in \mathcal{W}^+$ を変換結果として出力する．この際，以下の式が示すように，単語 w を入力記号列 $\boldsymbol{y}$ の組 $u = \langle w, \boldsymbol{y} \rangle$ を単位とする言語モデルによる生成確率を評価基準とする．

$$\begin{aligned} \underset{w}{\mathbf{argmax}}\, P(\boldsymbol{w}|\boldsymbol{y}) &= \underset{w}{\mathbf{argmax}} \frac{P(\boldsymbol{w}, \boldsymbol{y})}{P(\boldsymbol{y})} \\ &= \underset{w}{\mathbf{argmax}} P(\boldsymbol{u}) \end{aligned}$$

ここで単語列 $\boldsymbol{w}$ は表記文字である．$P(\boldsymbol{u})$ は，u を単位とする n-gram モデルを用いて，以下のようにモデル化される．

$$P(\boldsymbol{u}) = \prod_{i=1}^{h} P(u_i|\boldsymbol{u}_{i-n+1}^{i-1})$$

$$P(u_i|\boldsymbol{u}_{i-n+1}^{i-1}) = \begin{cases} P(u_i|\boldsymbol{u}_{i-n+1}^{i-1}) & (u_i \in \mathcal{U}) \\ P(\texttt{UU}|\boldsymbol{u}_{i-n+1}^{i-1}) M_{y,n}(\boldsymbol{y}_i) & (u_i \notin \mathcal{U}) \end{cases}$$

ここで $\mathcal{U}$ は言語モデルの語彙 (単語と入力記号列の組の集合) を表す．この式の中の u_i $(i \leq 0)$ と u_{h+1} は，文頭と文末に対応する記号 BT であり，UU は未知の組を表す記号である．また，$\boldsymbol{y}_i = y(u_i)$ は $u_i = \langle w_i, \boldsymbol{y}_i \rangle$ の入力記号列である．

式 (1) の $P(u_i|\boldsymbol{u}_{i-n+1}^{i-1})$ と $P(\texttt{UU}|\boldsymbol{u}_{i-n+1}^{i-1})$ は，語彙に BT と UU を加えた $\mathcal{U} \cup \{\texttt{BT}, \texttt{UU}\}$ 上の n-gram モデルである．パラメータは，単語に分割されかつ入力記号列が付与されたコーパスから以下の式を用いて最尤推定する．

$$P(u_i|\boldsymbol{u}_{i-n+1}^{i-1}) = \frac{N(\boldsymbol{u}_{i-n+1}^{i})}{N(\boldsymbol{u}_{i-n+1}^{i-1})}$$

ここで，$N(\boldsymbol{u})$ はコーパス中の表記と読みの組列 $\boldsymbol{u}$ の出現回数を表す．また，$M_{y,n}(\boldsymbol{y}_i)$ は入力記号列の n-gram モデルによる未知語モデルである．

3 仮名漢字変換クライアント

本章では仮名漢字変換クライアントについて説明する．仮名漢字変換クライアントはテキストエディタである Emacs 上で動作し，仮名漢字変換サーバとプロセス間通信によりデータをやり取りする．なお，学術論文執筆の特性上 Emacs 上で動作するクライアントでシステムを構築したが，本論文の仮名漢字変換サーバのクライアントとして文献 [5] のウェブブラウザ上で動作するクライアントにも対応している．

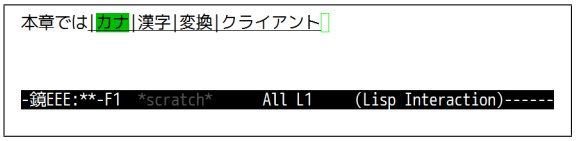

図 1: 変換結果を表示するクライアント

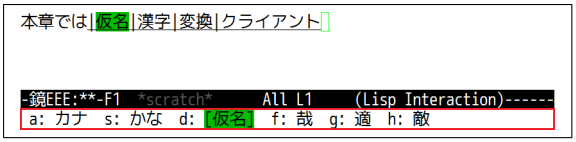

図 2: 変換候補を提示するクライアント

クライアントの主な動作は，ユーザーからのローマ字入力を平仮名に変換して仮名漢字変換サーバに送信し，サーバから返ってくる変換結果をユーザーに提示することである．図 1 はクライアントが変換結果を表示している画面である．サーバからの変換結果は単語分割済みの仮名漢字混じり列であり，クライアントは縦棒 “|” を用いて単語境界を表す．

本クライアントでは，単語ごとの変換候補を表示することができる．クライアントはユーザーから別の変換候補が求められると，サーバに候補の列挙を要求し，その返答を Emacs 下部のミニバッファに表示する．図 2 の最下段 (赤枠) 部分にこの動作を示す．これにより，最初の変換結果が誤っている場合，ユーザーは別の変換候補を表示することができるようになり，ユーザーの求める単語を最終的に選択できるようになる．また，表示されている単語分割境界を変えることができる．単語の分割位置が誤っている場合などに，分割位置を変更し，その分割位置で単語の変換候補を表示することができる．

本クライアントは，仮名漢字変換結果をユーザーに提示するだけでなく，変換ログをファイルに出力することで，ログを収集することができる．仮名漢字変換ログは，自動単語分割の精度向上 [5] などのように、自然言語処理に活用することができる．

表 2: 確率的タグ付与のための学習コーパス

学習コーパス	文数	単語数	文字数
BCCWJ Core	56,753	1,324,951	1,911,660
日経新聞	8,164	240,097	361,843
和英辞書の例文	11,700	147,809	197,941
辞書		単語数	
UniDic			234,652
姓名			197,552
数字			280

4 評価実験

この章では，仮名漢字変換の実験の結果とそれによる評価について述べる．また，別の分野のテキストを学習コーパスに入れることの影響についても実験的に考察する．執筆の対象とする分野としては，自然言語処理と医薬学の論文とした．そのため，言語モデルの学習コーパスには言語処理の論文と医学書を用いた．

4.1 実験の設定

実験の設定について順に説明する．

4.1.1 コーパス

表 1 に実験で使用する仮名漢字変換の学習コーパスとテストコーパスを示す．BCCWJ-train と BCCWJ-test は，現代日本語書き言葉均衡コーパス (BCCWJ)[6] の人手でアノテーションされた学習コーパスである．BCCWJ-nonc は，BCCWJ のアノテーション情報のない NonCore データに対して，表 2 で学習したモデルにより決定的に単語分割，及び読み推定したコーパスである．BCCWJ-*は，一般的なドメインのコーパスとして用いる．NLP-train は，言語処理学会 20 周年特別企画において公開された言語処理学会年次大会予稿集と言語処理学会論文誌の抽出テキスト[1]から，無作為に選択した 43,173 文に，擬似確率的タグ付与したコーパスである．NLP-test は，情報処理学会自然言語処理研究会の予稿原稿である文献 [7] の 265 文に対して人手でアノテーションしたコーパスである．NLP-*は，自然言語処理ドメインのコーパスとして用いる．MMH-*は，医学ドメインのコーパスとして，Web 上のメルクマニュアル医学百科[2]から抽出したテキスト 1,000 文に人手でアノテーションしたコーパスが MMH-test であり，50,915 文に擬似確率的タグ付与したコーパスが MMH-train である．擬似確率的タグ付与のために単語境界確率・読み確率を求めるモデルは，表 2 に示すコーパスを用いて学習した．

4.1.2 語彙

人手でアノテーションされたコーパスである BCCWJ-train の語彙は，全ての単語を語彙に含める．一方で，自動タグ付与された単語，特に擬似確率的タグ付与によりアノテーションされた単語は，誤った単語

[1] http://nlp20.nii.ac.jp/resources/
[2] http://mmh.banyu.co.jp/

表 3: 仮名漢字変換精度 (F 値)

モデル	学習コーパス				テストコーパス		
	BCCWJ-train	BCCWJ-nonc	NLP-train	MMH-train	BCCWJ-test	NLP-test	MMH-test
G--	o	o			93.70	89.45	93.55
GN-	o	o	o		93.69	96.33	93.54
G-M	o	o		o	93.71	89.71	97.08
GNM	o	o	o	o	93.73	96.52	97.10

を含む．このため，BCCWJ-nonc の語彙は BCCWJ-nonc に含まれる単語のうち UniDic に含まれる単語の集合とし，確率的タグ付与コーパスの語彙は単語を頻度で降順に並べた上位 3 分の 2 の単語集合とした．そして，BCCWJ-train，BCCWJ-nonc，確率的タグ付与コーパスの語彙の和集合を仮名漢字システムの語彙とした．

4.1.3 モデル

前述のコーパスを用いて，比較する対象として，基本となる BCCWJ-train と BCCWJ-nonc に加えて，各分野の学習コーパスを利用する場合としない場合の 4 通りについてモデルを学習した (表 3 参照)．

4.2 評価

4.2.1 評価方法

各テストコーパスをそれぞれのモデルで 1 文単位で仮名漢字変換した際の変換精度を比較し評価を行う．変換精度は，自動変換結果とテストコーパスを比較することで得られる文字単位の再現率と適合率とそれらの調和平均 (F 値) とした．

4.2.2 評価実験

結果を表 3 に示す．GNM が，全てのテストコーパスに対して高い精度を示した．この結果は，それぞれのドメインごとにコーパスを選択してモデルを学習するよりも，あらゆるドメインのコーパスを用いてモデルを学習する方が精度が高いことを示唆する．つまり，様々な学術分野のコーパスを加えた言語モデルに基づく仮名漢字変換サーバを作る意義があると言える．

また，G--の精度を比較すると，NLP-test に対する精度が低い．これは自然言語処理ドメインの仮名漢字変換が困難であることを示す．これに対して，NLP-train をモデルに追加すること (GN-,GNM) で 6%以上の精度向上が見られた．NLP-train は実験のため約 5 万文程度を用いているが，公開されている全ての自然言語処理論文約 70 万文を用いてモデルを学習すれば，さらなる精度向上が期待できる．

5 おわりに

本論文の仮名漢字変換システムは学術論文執筆のために構築したシステムであり，我々はこのシステムを配布している[3]．共著者の一人は本論文で述べる設計の仮名漢字変換システムを 5 年以上常用しており，この論文の一部もそのシステムを用いて執筆されている．意思決定において意見を伺うメールにて「語彙論はありませんか」と問うたことはあるが，論文執筆においてはすこぶる便利である．

実験のように自然言語処理の論文に対する仮名漢字変換精度が非常に高く，執筆効率が上がると考えられるので是非活用されたい．

参考文献

[1] 森信介, 土屋雅稔, 山地治, 長尾真. 確率的モデルによる仮名漢字変換. 情処論, Vol. 40, No. 7, pp. 2946–2953, 1999.

[2] Shinsuke Mori, Daisuke Takuma, and Gakuto Kurata. Phoneme-to-text transcription system with an infinite vocabulary. In *Proc. of the COLING06*, 2006.

[3] 森信介, 笹田鉄郎, Neubig Graham. 確率的タグ付与コーパスからの言語モデル構築. 自然言語処理, Vol. 18, No. 2, 2011.

[4] Graham Neubig, Yosuke Nakata, and Shinsuke Mori. Pointwise prediction for robust, adaptable Japanese morphological analysis. In *Proc. of the ACL11*, pp. 529–533, 2011.

[5] 高橋文彦, 森信介. 仮名漢字変換ログを用いた単語分割の精度向上. 言語処理学会第 21 回年次大会予稿集, 2015.

[6] Kikuo Maekawa. Balanced corpus of contemporary written Japanese. In *Proc. of the 6th Workshop on Asian Language Resources*, pp. 101–102, 2008.

[7] 森信介, 小田裕樹. 3 種類の辞書による自動単語分割の精度向上. 情報処理学会研究報告, 第 NL-193 巻, 2009.

[3] http://plata.ar.media.kyoto-u.ac.jp/takahasi/kagami-emacs/