正規-崩れ文字列アライメントと文字種変換を用いた崩れ表記正規化に基づく日本語形態素解析

斉藤 いつみ　貞光 九月　浅野 久子　松尾 義博
NTT メディアインテリジェンス研究所
{saito.itsumi, sadamitsu.kugatsu, asano.hisako, matsuo yoshihiro}@lab.ntt.co.jp

1　はじめに

近年 Twitter 等を代表とするマイクロブログが普及し，個人によって書かれたテキストを対象とした評判分析や要望抽出，興味推定に基づく情報提供など個人単位のマーケティングのニーズが高まっている．一方このようなマイクロブログ上のテキストでは口語調や小文字化，長音化，ひらがな化，カタカナ化など新聞等で用いられる標準的な表記から逸脱した崩れた表記が多く出現し，新聞等の標準的な日本語に比べ形態素解析誤りが増加する．ここで，次の例文を考える．

・映画を見ようそぉしよう！

上記の文に対して，Mecab[2] を用いて解析を行うと以下のような解析結果となる．

映画 (名詞)/を (助詞)/見よ (動詞・命令)/うそ (名詞)/ぉしよう (名詞)/！(記号)/

上記の例の場合,「ぉ」という崩れた表記によって「そう（副詞)」という正しい形態素の解析に失敗するだけではなく「うそ（名詞)」という誤った単語の湧き出しも生じている．上記の例について「ぉ」を「う」に変換して解析を行った場合の解析結果は以下のようになり正しく解析される．

映画 (名詞)/を (助詞)/見よ (動詞・未然)/う (助動詞)/そう (副詞)/しよ (動詞)/う (助動詞)/！(記号)/

従来はこれらの崩れ表記に対し，崩れた表記を直接辞書に追加する方法 (例えば,「そぉ(副詞)」を辞書に追加する) や，事前に人手で定めたルールに基づき崩れた表記を新聞等の表記に正規化して解析する方法 [5, 4](例えば,「ぉ→う」というルールを作成し，崩れた文から正規化された文への書き換えを行う，正規文字列を展開して辞書引きを行い形態素ラティスを拡張する) などが用いられてきた．しかし，人手による辞書追加やルール作成で現実に存在する多くのパタンを網羅することは高コストであり，再現率の点で限界がある．また，むやみに辞書に追加することによる解析悪化なども存在するため，既存手法では扱える現象が限られてしまうという課題があった．

本研究では，基本的には従来法 [5, 4] と同様の文字列正規化パタン (「ぉ→う」等) を用いて正規文字列を展開し形態素ラティスを拡張するという手法を用いるが，(1) ルールを人手で作成するのではなく正規表記と崩れ表記のアライメントから統計的に求める，(2) 異文字種展開と組み合わせる，(3) 識別モデルによる定式化，によって精度の向上を試みる．表 1 には，崩れ表記の分類と本研究で扱う範囲（網掛け部）を示した．対象範囲は，音的な類似という点で特定のパタンが存在すると考えられる口語調の崩れ表記や，異表記（小文字化，同音異表記，ひらがな化，カタカナ化）とした．これらを対象とした理由は，崩れ表記全体の中で占める割合が大きいとともに今回の提案手法で統一的に表現できる現象であったためである．

表 1: 崩れ表記の分類と本研究の対象範囲

大分類	小分類	具体例
口語	促音の挿入，置換	かっわいい, いこっか
	長音の挿入，置換	ねむーーい, とーきょー
	母音の挿入	まあるく, きたあああああ
	発音の崩れ	すっげぇ, くだしゃい
異表記	小文字化	ぃぃよ,ぉうち
	カタカナ/ひらがな化	アリガトウ, てすと
	同音異表記	まぢ, 少しづつ
	形の類似	ネ申
誤字脱字	タイプミス	これをを
	送り仮名誤り	面倒臭せ
	漢字の誤用	待ち通しい
略語	略語	おめ, よろ
方言	方言	やらへん, してんねや

本研究の全体構成を図 1 に示す．本研究のポイントは正規化の候補展開と識別モデルによる定式化であり，正規化候補展開については 2 節で，識別モデルの定式化については 3 節で詳しく述べる．実験結果は 4 節で示し，5 節でまとめと今後の課題を示す．

2　文字列正規化パタンの抽出と候補展開

2.1　正規-崩れ表記ペアのアライメントに基づく文字列正規化パタン抽出

文字列正規化パタンの抽出について，人手であらゆるパタンを記述することはコストが高く個々のパタンの起こりやすさを考慮することが難しい．本研究では，文字列の変化パタンを統計的に抽出するため，正規-崩れ表記の正解ペアデータを用いて正規化パタンの推定を行う．正解ペアデータの作成に際しては，まず崩れ

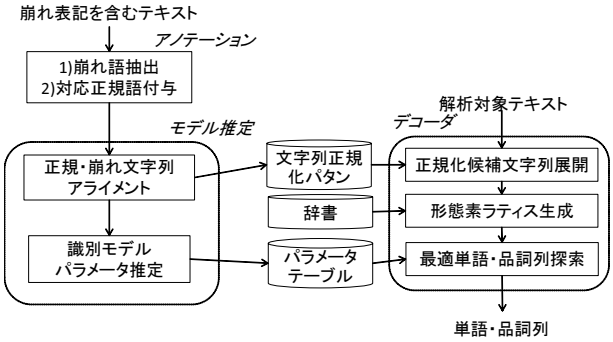

図 1: 提案手法の全体構成

表記を含むテキスト（ブログや Twitter）から崩れ表記（例:おぃしーぃ）を人手で抽出した．そして，抽出した崩れ表記に対して人手で正規表記（辞書に存在する表記）を付与する（おぃしーぃ→おいしい）．正解ペアの例を表 2 に示した．これらのペアデータ（おぃしーぃ，おいしい）から文字列アライメントの推定に基づき文字列レベルの正規化パタンを統計的を抽出する．(例：ー→ null，ぃ→い)．本研究では複数文字アライメントを扱える Haizhou ら [1] で示された次式を用いてアライメントを推定する．(本研究ではユニグラム確率を考える)

$$p(s,w) = \max \prod_{k=1}^{K} p(c_{sk}, c_{wk}) \tag{1}$$

ここで，s は崩れ文字列，w は正規文字列を表し，c_{sk}，c_{wk} はそれぞれ s，w 中の部分文字列を表す．K はアライメントにおける部分文字列の個数を表す．以下に具体的な推定手順を示す．

1. step1:初期確率値設定
2. step2:期待値の計算
   全てのテキストについてアライメントを求めた結果を用いて，文字列変換確率を次式に従って求める．ただし，$N(c_{sk}, c_{wk})$ は全テキスト中で部分文字列アライメント (c_{sk}, c_{wk}) が出現した回数である．

$$p(c_{sk}, c_{wk}) = \frac{N(c_{sk}, c_{wk})}{\sum N(c_{sk}, c_{wk})} \tag{2}$$

3. step3:経路確率とアライメントの計算
   ステップ 2 で求めた文字列変換同時確率 $p(c_{sk}, c_{wk})$ を用いて，もっとも生起確率が高い経路（アライメント）を新たな最適アライメントとして求める．図 2 に経路探索の計算例を示した．計算時には，-log$p(c_{sk}, c_{wk})$ を各アライメントのコストとみなしコストが最小となる経路を選択する．
4. step4:繰り返し
   アライメントが収束するまでステップ 2，3 を繰り返す．

表 2: 崩れ表記抽出と正規表記付与の例

崩れ表記	正規表記
ありがとうございま〜す	ありがとうございます
いちお	いちおう
うっつらうっつら	うつらうつら
お願いしま〜す	お願いします
くださ〜い	ください
じゃあない	じゃない
ずーーーっと	ずっと

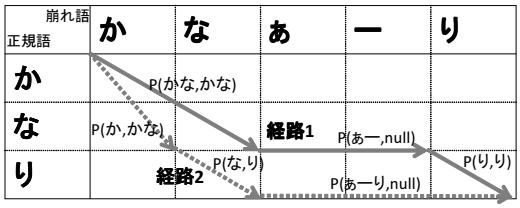

図 2: 正規表記と崩れ表記の文字列アライメント計算例

2.2 正規化候補展開

デコード時には 2.1 で抽出した文字列レベルの正規化候補展開を行った上で辞書引きを行い，正規語の展開を行う．この際，文字列正規化の他に文字種正規化候補展開も行う．図 3 に文字列正規化候補展開，文字種正規化候補展開と形態素ラティス生成の例を示した．以下にそれぞれの詳細を述べる．

2.2.1 文字列正規化候補展開

まず事前正規化として,「っ」と「ー」の連続に関しては 1 文字まで縮約させる．また，母音の連続に関しては 3 文字まで縮約させる．これらの事前正規化については，日本語の特徴を踏まえ意味変化が生じない範囲で定めた．次に，正規-崩れ文字列アライメントモデルで抽出された変換テーブル（文字列正規化パタン）を用いて，崩れ文字列を入力文にマッチさせ，マッチした箇所について正規文字列を展開し，辞書引きの際には展開した文字列に対しても辞書引きを行った．

2.2.2 文字種正規化候補展開

Twitter などのテキストでは，標準的な表記が漢字やカタカナの単語に関してひらがな表記される，あるいは標準的な表記が漢字やひらがなの単語がカタカナで表記される場合が多く存在する．これらの現象に関しては文字列アライメントから文字列単位で候補生成を行うことは非効率であり，既存の辞書の読みを利用して辞書引き時に候補展開を行うことを考える．例えば,「バイト」,「教科書」,「ありがとう」といった表記が辞書にあった場合，入力文に「ばいと」,「きょうかしょ」,「アリガトウ」といった表記があれば辞書の読みを利用して「バイト」,「教科書」,「ありがとう」をそれぞれ展開する．文字列正規化候補展開と文字種正規化候補展開を組み合わせることで,「カワィィ」→「かわいい」,「かぅんたー」→「カウンター」といった

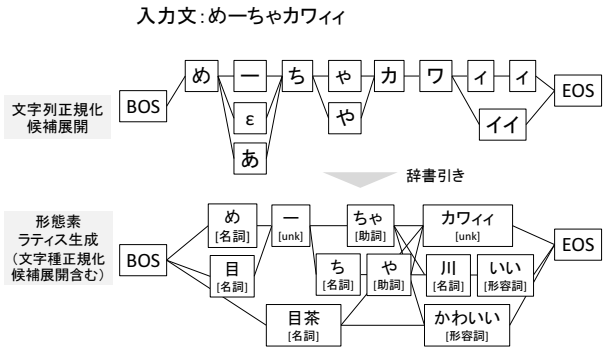

図 3: 文字列正規化候補展開と形態素ラティス生成の例

文字列アライメントだけではカバーできないより複雑な崩れ表記に対する正規化候補の展開も行うことが可能になる．

3 識別モデルに基づく定式化

2 節で示したように入力文から文字列や文字種を展開して形態素ラティスを生成する場合，不要な候補も多く生成されるため既存の形態素コストや品詞連接コストをそのまま用いると解析誤りが増加するなどの悪影響が考えられる．精度を向上させるためには生成した形態素ラティスに対し適切な重み付けを行うことが必要となる．本研究では文字列や文字種といった多様な崩れ表記を対象としているため，多様な素性を柔軟に考慮することができる識別的モデルを用いて表現することにする．ここである入力文に対して最適な出力 $\hat{\mathbf{w}}$ を計算する問題は以下のような線形モデルの最小化として定義できる．

$$\hat{\mathbf{w}} = \arg\min_{\mathbf{w}} \sum_k \lambda_k \phi_k(\mathbf{w}) \tag{3}$$

ここで，$\mathbf{w}$ は単語表層+品詞列，λ_k は k 番目の素性に対する重み，$\phi_k(\mathbf{w})$ は k 番目の素性関数を表す．今回は素性として品詞連接コスト $h(t_{i-1}, t_i)$，単語コスト $h(w_i, t_i)$，単語表層 n-gram コスト $-\log p(w_i|w_{i-1})$，表記崩れコスト $-\log p(c_{wk}|c_{sk})$，文字 n-gram コスト $g(\theta)$，文字種変換コスト（ひらがな変換コスト，カタカナ変換コスト）の 7 つを用いた．このうち，品詞連接コスト $h(t_{i-1}, t_i)$，単語コスト $h(w_i, t_i)$ は Mecab で提供されている推定値を用いた．ここで，t_i は i 番目の品詞，w_i は i 番目の単語を表す．単語表層 n-gram コスト $-\log p(w_i|w_{i-1})$ はブログの形態素正解が付与されていないラベルなしコーパスの自動解析結果より求めた．文字 n-gram コスト $g(\theta)$ はブログと新聞のテキストを用いて計算した．文字列正規化コスト $-\log p(c_{wk}|c_{sk})$ については，2.1 節の文字列アライメントモデルで得られた推定値を用いている．ひらがな化コストとカタカナ化コストについては，それぞれ入力文がひらがな，カタカナ表記で辞書の読みを用いて候補展開を行った場合に付加されるコストである．例えば,「ばいと」という表記から「バイト」を展開した場合にはひらがな変換コストを付加する．ここで，c_{sk} , c_{wk} はそれぞれ崩れ文字列，正規文字列を表す．(3) 式の重みパラメータ λ_k の推定については次の (4),(5) 式に示した目的関数を用いた MERT（誤り率最小化学習）[3] により行った．目的関数として，今回は正規文と崩れ文の形態素解析結果の異なり数を用いた．

表 3: 獲得された文字列正規化パタン例

c_{sk}	c_{wk}	$p(c_{wk}\|c_{sk})$	c_{sk}	c_{wk}	$p(c_{wk}\|c_{sk})$
ねー	ない	0.41489	ヴァ	バ	0.39024
ナィ	ない	0.57143	パ	バ	0.00377
ね〜	ない	0.03846	ぃ	い	0.46593
ねェ	ない	0.66667	ーぃ	い	0.83333
ね	ない	0.01830	ゆ	い	0.19250
ねえ	ない	0.55128	っ	い	0.00445
ん	な	0.00334	ー	い	0.01519
にゃ	な	0.22430	げー	ごい	0.54839
ウ	う	0.00917	っげー	ごい	0.66667
ゆ	う	0.00250	む	むい	0.00664
ぅ	う	0.37971	みぃ	むい	0.23077
お	う	0.01575	ァ	あ	0.03008
おおお	う	0.16667	ー	あ	0.01113

$$\hat{\lambda} = \arg\min_{\lambda \in \Lambda} l(\mathbf{w}; \lambda) \tag{4}$$

$$l(\mathbf{w}; \lambda) = \sum_{i=1}^{N} \mathrm{error}(E, \arg\max_{\mathbf{w} \in A} \sum_k \lambda_k \phi_k(\mathbf{w})) \tag{5}$$

4 評価と考察

4.1 実験データと文字アライメント結果

モデル推定に使用したデータはブログ，Twitter データから収集した崩れ表記と対応する正規表記のペアデータ約 17000 ペアである．アライメント計算の結果，得られた文字列崩れパタン数は 1956 種類であった．表 3 に獲得された文字列崩れパタンの例を示す．「ない」や「たい」といった正規文字列に対しては多くのパタンが獲得できた．また,「ヴァ→バ」といった音の類似や「ぅ→う」（小文字化）,「ー→う」（長音化）といった従来のルールベースで用いられている代表的なルールのほかにも,「お→う」,「っ→い」,「にゃ→な」といった低頻度な崩れ表記に関しても獲得可能であることが確認できた．単語表層 n-gram モデルの構築には，ブログの形態素正解ラベルなしデータを約 824 万を Mecab を用いて自動解析した結果を使用した．評価データは Twitter データから崩れ箇所を 1 箇所以上含む 300tweets を人手で抽出して作成した．

4.2 評価方法と比較手法

本研究では，評価文の崩れ箇所に対し 1) 文字列正規化の従来手法 [5] を実装した結果，2) 提案手法の文字列正規化のみを実装した結果，3) 提案手法の文字列正規化と文字種正規化の双方を実装した結果の 3 つの解析結果について比較した．解析結果の集計について

表 4: 解析結果差分の比較評価

	A	B	C	D	E
文字列正規化（従来）	68	46	3	134	61
文字列正規化（提案）	240	37	28	163	133
文字列正規化＋文字種正規化 (提案)	**267**	**37**	**28**	**140**	**131**

表 5: 提案手法による解析結果の改善例

	解析結果例
改善例 1	分かるょそれ〜
Mecab	分かる (動詞)/ょそれ (名詞)/〜(名詞)/
提案法	分かる (動詞)/よ (助詞)/それ (名詞)/〜(名詞)/
改善例 2	ぢゃーあたしもクレジットにしてみよっかな
Mecab	ぢ (名詞)/ゃ(名詞)/ー (名詞)/あたし (名詞)/ も (助詞)/クレジット (名詞)/に (助詞)/し (動詞)/ て (助詞)/み (動詞)/よっ(動詞)/か (助詞)/な (助詞)/
提案法	じゃ(接続詞)/あたし (名詞)/も (助詞)/ クレジット (名詞)/に/し (動詞)/て (助詞)/ みよ (動詞)/う (助動詞)/か (助詞)/な (助詞)/
改善例 3	らぁめん美味しい
Mecab	ら (名詞)/ぁめん (名詞)/美味しい (形容詞)/
提案法	ラーメン (名詞)/美味しい (形容詞)

は，それぞれの手法の解析結果と正規化を行わない場合（通常の mecab で解析した結果）との差分を抽出し，差分を A : 誤って解析されていた箇所を正規化して正しく解析，B : 正しく解析していた箇所を正規化して正しく解析（なー（終助詞）→な（終助詞）など，意味が同じでいずれも正しく解析されている場合），C : 正しく解析していた箇所を誤って解析，D : 誤って解析されていた箇所を誤って解析（正規化可能な崩れ未知語），E : 誤って解析されていた箇所を誤って解析（正規化不可能な未知語）の 5 つに分類した．

4.3 実験結果

表 4 の集計結果より，提案手法の文字列正規化と文字種正規化を組み合わせた場合が最も多く正規化に成功していることがわかる．文字列正規化のみと文字列正規化＋文字種正規化の結果を比較すると，文字種正規化による正規化成功数は約 10 %増加しており，文字列正規化による効果が最も大きいが文字種正規化を組み合わせることでさらに正規化可能な形態素数を増加させられることがわかった．解析誤りの増加についても文字種正規化を追加したことによる影響は大きくなく，文字列正規化に失敗していることによる解析誤りが主な誤りの内訳であるといえる．既存手法の単純なルールを用いた場合と比較すると，正規化成功数が約 4 倍であり崩れ表記のカバー率を大きく向上させることができた．既存手法では解析誤りの増加は最も少ないが，提案手法の解析誤り改善数が解析誤り増加数に比べ大きく上回っているため，全体でみると提案手法が最も精度を向上させることができた．

4.4 考察

表 5 には解析結果の改善例を示した．改善例では，小文字化や異表記（音の類似），発音の崩れ，文字種変換について正しく正規化して解析できていることがわかる．今回対象とした正規化可能な崩れ表記に関する失敗例主に (1) 探索誤り，(2) 候補列挙ができないことによる誤り，(3) 枝刈りによる誤りが存在した．探索誤りに関しては，正しい候補が列挙できたが選択されなかったもので助詞抜けとひらがな化が連接している文など twitter 特有の文法が影響しており，品詞連接コスト等の改善がや文字列変換確率の精緻化などが必要になる．候補列挙ができない例については，アライメントの正解ペアの自動獲得や既知ルールから類似パタンを生成するなどの手法が必要となる．枝刈りミスについては，速度と精度の双方を満足するためのより効率的な探索方法の検討が必要になる．また，提案法でも既存解析器でも誤った例として真の未知語（特にひらがなの固有名詞や感動詞）が挙げられる（表 4 の E にあたる部分）．これらの問題に対しては正規化に基づくアプローチとは異なる未知語処理の手法が必要となる．

5 まとめと今後の課題

本研究では正規表記と崩れ表記の正解ペアデータから抽出した文字列正規化パタンと文字種正規化候補展開を解析器に組み込み，従来扱えていなかった崩れ表記まで解析が可能になることを示した．この際，多様な素性関数をもちいた識別モデルの枠組みでコストを再定義することにより解析誤りの増加を抑えつつ崩れ表記の解析を改善させることができた点が本研究の成果である．ただし，正規化による解析悪化による意味変化が生じている部分もあり，これらは少数であってもその後の解析に悪影響を及ぼすと考えられるため，モデルの精度向上に取り組んでいく必要がある．また，正規化ルールを自動拡張する手法や解析精度と速度の双方を向上させるための探索アルゴリズムについても検討を行っていく予定である．

参考文献

[1] Li Haizhou, Zhang Min, and Su Jian. A joint source-channel model for machine transliteration. *Proceedings of the 42Nd Annual Meeting on Association for Computational Linguistics*, 2004.

[2] T. KUDO. Mecab : Yet another part-of-speech and morphological analyzer. *http://mecab.sourceforge.net/*, 2005.

[3] Machery W, Och F J, and Thayer I Uszkoreit J. Lattice-based minimum error rate training for statistical machine translation. *In Proc. of EMNLP*, Vol. 1, pp. 725–734, 2008.

[4] 岡照晃, 小町守, 小木曽智信, 松本裕治. 表記のバリエーションを考慮した近代日本語の形態素解析. 人工知能学会全国大会講演集, 2013.

[5] 勝木健太, 笹野遼平, 河原大輔, 黒橋禎夫. Web 上の多彩な言語表現バリエーションに対応した頑健な形態素解析. 自然言語処理学会年次大会講演集, pp. 1003–1006, 2011.