RE-001

# くだけた表現を修正するための教師なし学習方式の提案と評価

池田　和史 †　　柳原　正 †　　松本　一則 †　　滝嶋　康弘 †

**あらまし**　ブログ上の文書には口語的な表現や特有の表記などのくだけた表現が多数含まれるため、一般の形態素解析器を用いても十分な解析精度を得ることはできない。くだけた表現は人手により辞書登録されることが一般的であるが、人的コストの大きさや専門的な知識を必要とすることが課題である。本稿ではくだけた表現を文語的な表現に修正するための教師なし学習手法を提案する。提案手法ではくだけた表現の修正候補文字列をくだけた表現の少ない文書から自動的に検索し、修正ルールを生成する。生成した多数の修正ルールから文脈に適した修正ルールを選択的に適用するために、検索結果における修正候補文字列の出現頻度、修正前後の文字列間における編集距離、修正前後の文の形態素解析結果の比較、を用いて修正ルールをスコアリングする手法を合わせて提案する。提案手法を実装し、従来手法との性能比較評価実験を行った。各手法を利用したときの未知語の出現率や文節区切りの正確さ、修正前後の文の意味変化を定量的に評価した。提案手法では従来手法と同程度の文節区切りの正確さを維持しながら、対象文書の未知語出現数を 30.3% 減少させることに成功した。これは従来手法における未知語減少数の 2 倍以上である。

## 1. まえがき

近年、インターネットの普及により、一般ユーザによる Web 上での情報発信の手段としてブログが注目されており、ブログを対象とした情報抽出や検索、ランキングなどに関する研究が盛んに行われている[1),2)]。しかし、ブログ文書には「うっそー」、「すごーい」のような口語的な表現や「ｶわいい」、「わたUわ」（「わたしは」と読む）のような特有の表記などの**くだけた表現**が多く含まれ、その多くは一般の形態素解析器では未知の語として扱われるため、十分な言語解析精度を得ることができないという問題がある。現在ではくだけた表現を人手により辞書登録することが一般的であるが、未知語の登録には品詞や活用形の登録、既存の辞書との互換性の維持など、言語処理に関する専門的なスキルを必要とし、人的コストが大きい点が問題となる。我々の経験では、1 人月あたり約 3 万種類の未知語登録が可能であるのに対し、ブログ 600 万文を著名な形態素解析器 MeCab[3)] を用いて解析したところ、約 65 万種類の未知語が検出されたことから、ブログ文書のくだけた表現を正しく解析することは困難といえる。

ブログ文書のくだけた表現の多くは文語的な表現からの派生であり、派生の仕方にはいくつかの傾向が見られる。例えば「ｶわいい」や「わたUわ」のように形状が似ている文字の代替が起こりやすい傾向にある。他の傾向として、「うっそー」や「すごーい」のように会話における発音の変化傾向に併せた表記がなされる。また、「かっこいい」がブログ上では「カッコイイ」と記載されるように、本来ひらがなで書かれるべき語を意図的にカタカナ表記にするなどの傾向がある。

これらのくだけた表現の解析精度を向上するため、本稿ではくだけた表現を文語的な表現へと修正する手法を提案する。提案手法ではくだけた表現の修正候補文字列をくだけた表現の少ない新聞コーパスなどから自動的に検索し、文字列変換のルール（**修正ルール**）を生成する。例えば、文字列「ｶわいい」を「かわいい」に変換することで、くだけた表現を文語的な表現に修正できる。生成した多数の修正ルールから文脈に適した修正ルールを選択的に適用可能にするため、検索結果における修正候補文字列の出現頻度、修正前の文字列から修正後の文字列への文字列編集距離、修正前後の文の形態素解析結果の比較、という 3 つの指標を用いて修正ルールをスコアリングする手法を合わせて提案する。

提案手法を実装し、従来手法と性能比較評価実験を行った。性能評価ではブログ 10 万文を評価対象とし、形態素解析時の未知語の出現率や文節区切りの正確さ、修正前後における文の意味の変化について定量的に評価した。提案手法では、従来手法と同程度の文節区切りの正確さで従来手法の 2 倍以上の未知語を減少させることに成功し、これは対象文書の未知語出現数の約 30.3% に相当する。また、修正ルールを適用するスコアの閾値を変化させることで、未知語数の減少と修正ルールの過剰適用という提案手法が持つトレードオフについても評価した。

## 2. 関連研究

チャットの口語的表現を対象とした形態素解析辞書拡張手法[4)] では、チャットの文章を分析し、人手によって辞書拡張のルールを作成することで、既存の辞書から派生した語を辞書登録する。例えば、「がっこう」は「がっこー」と表現されるなどの例から、直前の文字の母音が「o」の場合、「お、う、ー、〜」は互いに置換可能である、などのルールを提示している。この手法により、辞書登録の人的コストは軽減されるが、人手によるルール作成は作業者が参考にした文例に依

†KDDI 研究所：KDDI R&D Laboratories, Inc.

Unsupervised Approach to Modify Casual Sentences on Blog Documents.

Kazushi Ikeda, Tadashi Yanagihara, Kazunori Matsumoto, and Yasuhiro Takishima

存したり、主観に基づきやすい。以前の我々の研究報告[5]では文献 4) を参考にルールを作成し、ブログ 200 万文を形態素解析し、文節区切りに変化が見られた 53488 文のうち、600 文をサンプリングして評価したところ、37.2%の文はルール適用前と比べて文節区切りが悪化していることが確認された。

この他にも口語的表現や話し言葉を言語的な観点などから分析した形態素解析精度向上のための手法が提案されている。文献 6) では、「〜しちゃう」などの口語特有の言い回しを分析し、人手により辞書登録を行うことで、口語の形態素解析精度が向上することが報告されている。同様に、文献 7)，8) では、話し言葉の形態素解析を対象とした研究成果が報告されており、様々な観点から口語的表現を分析し、特徴を列挙している。しかし、上記のような言語解析には専門的なスキルや多くの労力を要するなど、人的コストの大きさが課題となる。また、形態素解析における未知語の解消についても盛んに研究が行われており、カタカナ語の表記の揺れを解消する手法[9]や Web から新語を獲得する手法[10]、未知語の品詞推定を行う手法[11]、単語分割境界を推定する手法[12]などが提案されている。これらの手法は未知語の解消に貢献するが、我々が対象としているくだけた表現の修正を対象としたものではない。

これに対し、我々が以前に提案した修正ルールの統計的スコアリグ手法[5]では、少数の汎用的な修正ルール（プリミティブルール）をあらかじめ人手により与える。プリミティブルールを元に、特定の文脈でのみ利用できる特殊な修正ルールを生成し、大規模コーパスを用いて統計的にスコアリングすることで、文脈に応じた修正ルールの選択を可能にした。例えば、「ゎ⇒は」と「ゎ⇒わ」という汎用的な修正ルールを人手により与えると、(a)「今日ゎ⇒今日は」や (b)「今日ゎ⇒今日わ」のような、より特殊な修正ルールを大規模コーパスから自動的に生成し、(a) の「今日ゎ⇒今日は」の方が統計的に正解率が高いことを学習する。しかし、この手法では与えられたプリミティブルールを組み合わせた修正ルールしか生成できないため、「困っちゃう」を「困ってしまう」に修正するためには「ちゃ⇒てしま」などの修正ルールを人手により与える必要があるなど、拡張性が低いことや修正結果が与えられたプリミティブルールに大きく依存してしまうという課題があった。

本稿で提案する手法は (1) くだけた表現の修正候補文字列をくだけた表現の少ない文書から自動的に検索し、修正ルールを生成するため、人手によるプリミティブルールの付与を必要としない点、(2) 検索結果における修正候補文字列の出現頻度と修正前後の文字列間における編集距離、修正前後の文の形態素解析結果の比較、の 3 つの指標を用いて修正ルールをスコアリングすることで、多数の修正ルールから文脈に適した修正ルールを選択的に適用可能な点、において文献 5) の手法とは大きく異なる。

## 3. 提 案 手 法

提案手法の全体像を図 1 に示す。提案手法では、くだけた表現を多く含むブログなどの文書を入力とし、くだけた表現

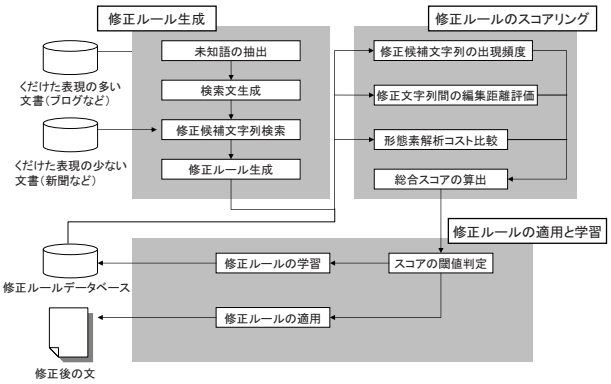


**図 1** 提案手法の全体像

の少ない新聞などの文書からくだけた表現の修正候補を自動的に検索し、修正ルールとして生成する。生成した修正ルールを (1) 検索結果における修正候補文字列の出現頻度、(2) 修正前後の文字列間における編集距離、(3) 修正前後の文の形態素解析コスト値の差分、によりスコアリングすることで、多数ある修正ルールの中から文脈に適した修正ルールを選択することができる。以下では、各処理の詳細について説明する。

### 3.1 修正ルールの生成

修正ルールの生成はくだけた表現の抽出と修正候補文字列の取得により実現される。ここでは「できるかどうゃゎ分かりません」というくだけた表現を含む文の修正を例に挙げて説明する（図 2）。くだけた表現の多くは形態素解析辞書に登録されていないため、形態素解析時に未知語として検出される（図 2 の (1)）。検出されたくだけた表現の修正候補文字列をくだけた表現の少ない新聞文書などから検索する。くだけた表現の未知語部分（図 2 では「ゃゎ」）を任意の文字列（ワイルドカード）とし、未知語部分に隣接する文字列と合わせて検索文とする（図 2 の (2)、ここでは隣接する 1 単語ずつを検索文生成に利用した）。くだけた表現の少ない文書から修正候補文字列を検索し、取得する（図 2 の (3)）。未知語部分から修正候補文字列への文字列変換を修正ルールとして生成する。（図 2 の (4)）。これにより、多数の修正ルールを自動生成できる。生成した多数の修正ルールのうち、文脈に適した修正ルールを選択的に適用ためのスコアリング手法について 3.2 節で説明する。

### 3.2 修正ルールのスコアリング

提案手法では修正ルールのスコアリングに、(1) 検索結果における修正候補文字列の出現頻度、(2) 修正前後の文字列間における編集距離、(3) 修正前後の形態素解析コスト値の差分、の指標を用いる。以下では各スコアの算出方法について説明する。

#### 3.2.1 修正候補文字列の出現頻度に基づくスコアリング

修正ルール生成時の検索結果における修正候補文字列の出現頻度を修正ルールのスコアリングに用いる。図 2 の (3) における検索結果の出現頻度をまとめると表 1 のようになる。出現頻度の高い文字列はくだけた表現が出現した文脈と類似

くだけた表現を含む文（修正対象となる文）：
「できるかどう ヵわ 分かりません」
形態素解析結果：
「できる/か/どう/ヵわ/分かり/ませ/ん」
未知語：ヵわ …(1)
検索文生成：
「どう*分かり」 …(2)
くだけた表現の少ない文書からの検索結果：
「これかどう かは 分かりません」
「よくあるかどう か 分かりません」
「どう したらいいのか 分かりません」
「この先どう かは 分かりません」 …(3)
「本当かどう か 分かりませんが」
「使うかどう かは 分かりませんけどね」
「あるかどう かは 分かりません」
「どう なっているか 分かりませんよ」
修正ルールの生成：
「ヵわ⇒かは」
「ヵわ⇒か」
「ヵわ⇒したらいいのか」 …(4)
「ヵわ⇒なっているか」

**図 2** 修正ルール生成方法の具体例

**表 1** 修正候補文字列の出現頻度に基づくスコアリング例

| 修正ルール | 出現頻度 | 出現頻度 / 検索件数 |
|---|---|---|
| ヵわ ⇒ かは | 4 | 0.5 |
| ヵわ ⇒ か | 2 | 0.25 |
| ヵわ ⇒ したらいいのか | 1 | 0.125 |
| ヵわ ⇒ なっているか | 1 | 0.125 |

した文脈でよく利用される表現であると考えられ、くだけた表現の修正候補文字列である可能性が高い。一方、類似した文脈であまり利用されていない表現は修正候補文字列である可能性が低い。評価値が検索件数に依存しないように、出現頻度を検索件数で割り、正規化して利用する。

#### 3.2.2 修正前後の文字列間における編集距離に基づくスコアリング

くだけた表現は文語的な表現から派生した表現であり、「すごーい」や「どぅかな」のように、文語的な表現に対して少数文字の挿入や削除、置換を行ったものであることが多い。ここで、文字列間の編集距離（レーベンシュタイン距離[13]）を考える。編集距離とは、二つの文字列がどの程度異なっているかを表す指標であり、一方の文字列を他方の文字列に変換するために必要な挿入、削除、置換の最小回数として与えられる。例えば、「フォーラム」から「ファーム」への編集は「ォ」を「ァ」に置換し、「ラ」を削除する方法が編集回数 2 回で最小となるため、編集距離は 2 である。

編集距離を用いると、図 2 で生成した各修正ルールのスコアは表 2 のようになり、「ヵわ⇒したらいいのか」や「ヵわ⇒なっているか」など、編集距離の大きい修正ルールはスコアが低くなる。また、くだけた表現では「ヤバい」や「カッコイイ」のように本来ひらがなで表記されるべき語がカタカナで表記されている例が多いことなどを考慮し、カタカナをひらがなに置換する編集距離を小さくするなどの重み付き編集距離を用いることも有効と考えられる。

**表 2** 編集距離に基づくスコアリング例

| 修正ルール | 編集手順 | 編集距離 |
|---|---|---|
| ヵわ ⇒ かは | 置換:2 回 | 2 |
| ヵわ ⇒ か | 置換:1 回、削除:1 回 | 2 |
| ヵわ ⇒ したらいいのか | 置換:2 回、挿入:5 回 | 7 |
| ヵわ ⇒ なっているか | 置換:2 回、挿入:4 回 | 6 |

#### 3.2.3 修正前後の形態素解析コスト値に基づくスコアリング

くだけた表現が出現する文脈における修正ルールの適応度合いを評価する指標として、形態素解析コスト値[14] を用いる。形態素解析コスト値とは単語の生起確率 (生起コスト) や単語同士の連接確率（連接コスト）などから算出される値で、複数ある単語区切り方のうち、もっともらしいものを選択する際に多くの形態素解析器で用いられる。提案手法では、修正箇所周辺における表現の自然さを推定する指標として、形態素解析コスト値を用いる。修正ルールの適用により、不自然な表現が生成された場合、表現周辺の生起コストや連接コストは大きくなることから、修正の誤りを推定する。

図 3 に形態素解析コスト値に基づくスコアの算出例を示す。各形態素における連接コストと単語生起コストの和を文頭からの累積で算出した値（累積コスト）の文末における値が文全体の単語区切りのもっともらしさを表すと考える。くだけた表現を含む「できるかどうヵわ分かりません」のような文は未知語部分の単語生起コストが大きいため、文全体の累積コストが大きくなる。修正ルール適用後の文の形態素解析コスト値をそれぞれ算出し、修正前の文の形態素解析コスト値との差分を修正ルールのスコアとする。一般的に短い文の方が形態素解析コスト値は小さくなる傾向にあるが、多数の文字列を削減するような修正ルールは 3.2.2 節の編集距離によるスコア値が低くなる。

#### 3.2.4 総合スコアの算出

修正ルールの総合的なスコア $score$ は修正候補文字列の出現頻度 $freq$、修正文字列間の編集距離 $dist$、形態素解析コスト値の差分 $cost$ を用いて一般的に (1) 式のように記述できる。ここで、関数 $f, g, h$ は各指標の重み付け関数である。本稿における実装では (2) 式のように単純にそれぞれ定数 $\alpha, \beta, \gamma$ とした。

$$score = f(freq) + g(dist) + h(cost) \quad (1)$$

$$score = \alpha \cdot freq + \beta \cdot dist + \gamma \cdot cost \quad (2)$$

4 節における実験で実際に生成した修正ルールを表 3 に示す。(2) 式における各スコア値が均等に影響するように重み付け定数を設定し、$\alpha = 1, \beta = -16, \gamma = -0.005$ とした。くだけた表現「てたょ」の修正例では出現頻度や形態素解析コストから、「てた」が最適な修正候補文字列であることを示している。「てたといえよう」なども修正候補文字列として得られているが、編集距離が大きいため選択されない。「今

**表 3** 修正例と各指標におけるスコア ($\alpha = 1, \beta = -16, \gamma = -0.005$)

| 修正例 | 出現頻度 (%) | 編集距離 | 形態素解析コスト | 総合スコア |
|---|---|---|---|---|
| てたょ⇒てた | 20 | 1 | -15757 | 83.3 |
| てたょ⇒てたよ | 0 | 1 | -14037 | 54.2 |
| てたょ⇒てたい | 2 | 1 | -10946 | 40.7 |
| てたょ⇒てたといえよう | 2 | 5 | -9108 | -32.5 |
| 今日ゎ午前⇒今日は午前 | 95 | 1 | -13421 | 146.1 |
| 今日ゎ午前⇒今日午前 | 0 | 1 | -13131 | 49.7 |
| 今日ゎ午前⇒今日だけ午前 | 0 | 2 | -6247 | -0.76 |
| ぉ金無い⇒金無い | 0 | 1 | -13131 | 49.7 |
| ぉ金無い⇒お金無い | 0 | 1 | -10974 | 38.9 |
| ぉ金無い⇒税金無い | 8 | 1 | -9887 | 41.4 |
| ぉ金無い⇒う金無い | 4 | 1 | -6654 | 21.3 |

```
修正前の文：　できるかどう ゎわ 分かりません
単語区切り：　できる/ か / どう /ゎわ/分かり/ませ/ ん
累積コスト：　5742/8263/11751/34685/39098/40388/39914
文全体のコスト：39914

修正候補１：　できるかどう かは 分かりません
単語区切り：　できる/ か / どう / か / は /分かり/ませ/ ん
累積コスト：　5742/8263/11751/14430/15438/19341/20631/20157
文全体のコスト：20157　修正前の文との差分（形態素解析コスト):-19757

修正候補２：　できるかどう か 分かりません
単語区切り：　できる/ か /どうか/分かり/ませ/ ん
累積コスト：　5742/8263/16737/20120/21410/20936
文全体のコスト：20936　修正前の文との差分（形態素解析コスト):-18978

修正候補３：　できるかどう したらいいのか 分かりません
単語区切り：　できる/か/どう/し/たら/いい/の/か/分かり/ませ/ん
累積コスト：　5742/8263/11751/ ...(略)... /26035/27325/26851
文全体のコスト：26851　修正前の文との差分（形態素解析コスト):-13063

修正候補４：　できるかどう なっているか 分かりません
単語区切り：　できる/か/どう/なっ/て/いる/か/分かり/ませ/ん
累積コスト：　5742/8263/11751/ ...(略)... /22975/24265/23791
文全体のコスト：23791　修正前の文との差分（形態素解析コスト):-16123
```

**図 3** 形態素解析コストを用いたスコアリング例

日ゎ午前」の修正例では出現頻度により、「今日は午前」が最適な修正であると判定する。「ぉ金無い」の修正例では「税金無い」なども高い総合スコアを得ているが、形態素解析コストが小さい「金無い」が文としてより一般的であると判定する。

### 3.3 修正ルールの適用と学習

スコアリングした修正ルールの適用と学習について説明する。修正ルールはスコアリングしているため、適用する閾値を設定することが可能である。閾値を低く設定した場合、より多くのくだけた表現を修正できるが、修正ルールの誤適用も増加する。閾値を高く設定した場合、くだけた表現の修正数は少なくなるが、誤適用を軽減することが可能である。修正ルールの適用数と誤適用のトレードオフについては 4 節で評価している。

提案手法では閾値以上のスコアを持つ修正ルールをデータベースに登録することで、修正ルールの再利用も可能である。修正候補文字列の検索において十分な候補数が得られなかった場合、データベースを参照することで、利用可能な修正ルールを取得することができる。また、くだけた表現に隣接する語によっては有用でない検索文が生成されることがある。例えば「子供はかゎいぃよね」という文の「ゎいぃ」が未知語であった場合、検索文は「子供は*ね」となり、任意の形容詞が修正候補文字列として得られる。多数の修正候補文字列に対してスコア値を算出するのは計算時間が大きく、適用誤りも起こりやすい。このような場合、データベースの修正ルールのみを用いて修正を行う。これはすでに別の文例で同じくだけた表現が正しく修正され、修正ルールがデータベースに登録されていることが期待されるためである。

## 4. 性能評価実験

提案手法を実装し、性能評価実験を行った。従来手法である、人手による**辞書拡張手法**[4] と修正ルールの**統計的スコアリング手法**[5] の 2 手法の性能と提案手法の性能を比較評価した。辞書拡張手法における課題としては辞書拡張ルールの過剰適用による単語区切りの誤りが挙げられる。統計的スコアリング手法における課題としては初期ルールを人手により与える必要があるため、修正可能な文数が少ないことが挙げられる。

これらの課題を考慮して、性能評価実験では未知語の出現率や単語区切りの正しさについて評価した。加えて、提案手法では修正ルールを教師なし学習するため、文の意味が変わるような修正ルールを自動生成する可能性があることから、文が持つ意味の変化についても評価した。また、提案手法における修正ルールを適用するスコアの閾値と未知語の出現率、過剰適用のトレードオフについても評価した。

### 4.1 実験の手順と環境

辞書拡張手法、統計的スコアリング手法、提案手法の 3 手法について、くだけた表現を含む文書の形態素解析を行い、下記の指標を評価した。(1) 修正された文に対する単語区切りが向上した文の割合（**単語区切り向上率**）、(2) 修正された文に対する単語区切りが悪化した文の割合（**単語区切り悪化率**）、(3) 修正された文に対する意味が変化した文の割合（**意味変化率**）、(4) 文書全体の形態素数に対する未知語数の割合（**未知語出現率**）。

単語区切りの向上とは、一般の形態素解析辞書（**基本辞書**）

**表 4** 意味変化の大小による修正の分類評価例

| 修正前 | 修正後 |
|---|---|
| (a) 意味変化が小さい修正例 | |
| 今日は猫ちゃん来てたょー | 今日は猫ちゃん来てた |
| とぉっても気持ちいい | とっても気持ちいい |
| おいしそぉだったんだけどね | おいしそうだったんだけどね |
| めっちゃ汗かいた | めっちゃ汗かいた |
| (b) 意味変化が大きい修正例 | |
| じゃあ、② 時に駅前で | じゃあ、七時に駅前で |
| ばかっぷる〜 | ばかっする〜 |
| 可愛いすぎぃ | 可愛すぎない |
| おっはよぉー | おっはよい |
| (c) 意味変化の有無が判定しにくい修正例 | |
| 来ぉへんよっ | 来へんよ |
| 私も遅くなる時ぁる∪ | 私も遅くなる時もある |
| ぜひぉためしあれ | ぜひためしあれ |
| ハハハよかったね | よかったね |

を用いた形態素解析において単語区切りに誤りを含む文が各手法によって正しく区切られる場合を指す。反対に、基本辞書では正しく単語区切りが行われていた文が各手法によって誤った単語区切りが行われた場合を悪化とする。単語区切りの正解判定は文献 4), 5) と同様に、文献 15) の手法を用いた。具体的には、人手で付与した正解の単語区切りと各手法における形態素解析結果の単語区切りを比較する。提案手法と統計的スコアリング手法では修正により、表層が変化するが、修正後の文の単語区切りが正しければ正解とする。

意味変化については各文を下記の 3 つの評価基準を用いて分類した。(a) 修正前後で意味はほとんど変化していない、(b) 修正前後で意味が明らかに変化している、または文の意味が理解できない、(c) (a),(b) の判断が付かない。各評価基準に分類される修正の例を表 4 に示す。(a) の意味変化が小さいと分類された修正では、与える印象はわずかに変化するかもしれないが、内容や事実関係に変化はないと考えられる。(b) の意味変化が大きいと分類された修正では、文の内容や事実関係に変化があったり、文が意味をなしていない。(c) の意味変化の有無が判定しにくい修正例では、文として不自然さはあるが、修正前と同じ意味と捉えられるものや、前後の文脈によっては意味が変わりうる修正などが含まれる。意味変化率の算出では (b) と (c) に分類される例を意味が変化したと判定した。上記の (1)〜(3) については、それぞれの手法で単語区切りまたは表層に変化のあった文のうち 600 文をサンプリングし、評価を行った。

以下に実験環境の詳細を示す。形態素解析器は MeCab[3] を用いた。従来手法と提案手法は共に人名などの固有名詞や流行語などが未知語として検出される場合を対象としていないことから、性能評価実験ではこれらの名詞 18 万語を追加登録した拡張 IPADIC 辞書を用いた。辞書拡張手法を評価するため、文献 4) を参考に辞書拡張ルールを作成し、機械的に形態素解析辞書に反映した。修正ルールの統計的スコアリング手法では人手により与えた 250 件の修正ルールをもとに学習用ブログデータ 1000 万文を用いて学習を行い、修正

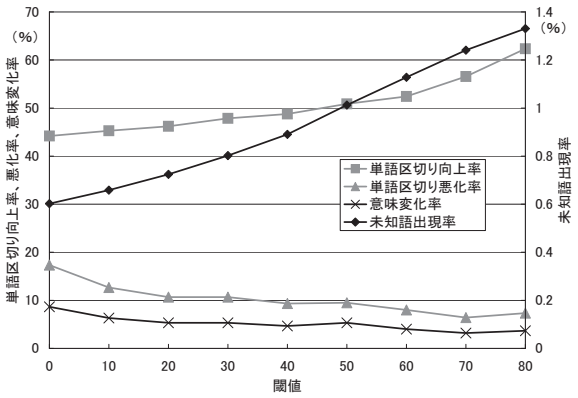


**図 4** 修正ルール適用の閾値と単語区切りの向上率、悪化率、意味変化率、未知語出現率

ルールをスコアリングした。

- 形態素解析器：mecab Version 0.97
- 形態素解析辞書（基本辞書）: mecab 標準 IPADIC 辞書に名詞 18 万語を追加
- プログラム実行環境: CPU 2.33 GHz 8 core、RAM 64GB、OS Linux version 2.6.24、 gcc version 4.1.2
- 統計的スコアリング手法の学習用文書: 商用ブログ 1000 万文
- 提案手法の修正候補文字列取得用文書: 毎日新聞（2007 年〜2008 年） 100 万文
- くだけた表現を含む評価対象文書: 商用ブログ 10 万文

### 4.2 実験結果

提案手法では修正ルールをスコアリングし、設定した閾値よりも高いスコアを持つ修正ルールを適用する。図 4 は閾値を 0 から 80 まで変化させたときの単語区切りの向上率と悪化率、意味変化率、未知語出現率を表している。閾値が小さいときは未知語を大幅に削減できているが、単語区切り悪化率や意味変化率が高いことから、削減された未知語の多くは形態素解析器上は正しく認識されていないことが分かる。閾値を高く設定するにしたがって、未知語出現率は上昇するが、単語区切りの正しさは向上し、意味変化率も小さくなることが分かる。

提案手法、辞書拡張手法、統計的スコアリング手法それぞれにおける単語区切りの向上率と悪化率、意味変化率、未知語出現率を表 5 に示す。ここでは提案手法における修正ルール適用の閾値を 60 に設定することで、人手により初期ルールを与える統計的スコアリング手法と同程度の単語区切りの向上率と悪化率、意味変化率となるように調整した。提案手法は辞書拡張手法と比べると単語区切り悪化率が低く、未知語出現率も小さい。提案手法における未知語出現率 1.128%は基本辞書の 1.619%に対して 30.3%低い。この未知語削減率は統計的スコアリング手法の 2 倍以上である。提案手法では教師なし学習により大規模な修正ルール集合を自動的に構築し、3 つの指標を用いて修正ルールをスコアリングすることで、高精度な修正ルールの適用を可能とする。これにより、従来手法と同程度の修正の正しさを維持しながら、より多くの未知語を削減することが可能となる。

**表 5** 各手法の性能比較

| 手法 | 単語区切り向上率 (%) | 単語区切り悪化率 (%) | 意味変化率 (%) | 未知語出現率 (%) |
|---|---|---|---|---|
| 基本辞書 | - | - | - | 1.619 |
| 辞書拡張手法 | 48.1 | 32.1 | - | 1.458 |
| 統計的スコアリング手法 | 52.1 | 9.7 | 4.0 | 1.377 |
| 提案手法 | 52.4 | 8.0 | 4.0 | 1.128 |

これらの実験結果から、提案手法は従来手法の課題であったルールの過剰適用やスケーラビリティの少なさといった問題を解決することができることを確認した。また、計算時間についても従来の統計的スコアリング手法はブログ 1000 万文による事前学習に 17 時間を要するのに対し、提案手法は事前学習を必要としない。修正に要する時間は提案手法で新聞コーパス量が 100 万文のとき、ブログ 1000 文を約 1 秒で修正することができ、従来手法と同程度の計算時間である。

## 5. ま と め

本稿ではブログ上の文書に多く見られる口語的な表現や特有の表記などのくだけた表現を文語的な表現へ修正する手法を提案した。提案手法ではくだけた表現の修正候補文字列をくだけた表現の少ない文書から自動的に検索し、修正ルールを生成する。生成した修正ルールを修正候補文字列の出現頻度、修正文字列間の編集距離、形態素解析コストの 3 つの指標を用いてスコアリングすることで、最適な修正ルールを適用することができる。

提案手法を実装し、従来手法である辞書拡張手法、統計的スコアリング手法と性能比較評価実験を行った。提案手法では従来手法と同程度の文節区切りの正確さを維持しながら、従来手法の 2 倍以上の未知語を解消できることを確認した。解消した未知語数は対象文書の未知語出現率の約 30.3% に相当する。加えて、提案手法における修正ルールを適用するスコアの閾値を変化させることで、未知語出現数の減少と修正ルールの過剰適用のトレードオフについても評価した。

## 参 考 文 献


1) 中島伸介，稲垣陽一，草野奉章：高信頼性情報の提示を目指した熟知度に基づくブログランキング方式の提案，日本データベース学会論文誌， Vol.7, No.1, pp.257–262 (2008).
2) 関口裕一郎，川島晴美，奥田英範，奥 雅博：ブログ発信者の特徴を利用した話題抽出手法，日本データベース学会論文誌， Vol.5, No.1, pp.9–12 (2006).
3) Kudo, T.: Mecab: Yet Another Part-of-Speech and Morphological Analyzer. http://mecab.sourceforge.net/.
4) 風間淳一，光石 豊，牧野貴樹，鳥澤健太郎，松田晃一，辻井潤一：チャットのための日本語形態素解析，言語処理学会第五回年次大会発表論文集，pp.509–512 (1999).
5) 池田和史，柳原 正，松本一則，滝嶋康弘：ブログ的表記を正規化するためのルール自動生成方式の提案と評価，日本データベース学会論文誌， Vol.8, No.1, pp.23–28 (2009).
6) 竹元義美，福島俊一：口語的表現を含む日本語文の形態素解析の実現と評価，情報処理学会自然言語処理研究会報告，pp.105–112 (1994).
7) 竹下 敦，福永博信：話し言葉に対する形態素解析，情報処理学会第 42 回全国大会，pp.1C–3 (1991).
8) 松本裕治，伝 康晴：話し言葉の形態素解析，情報処理学会音声言語情報処理研究会報告，pp.9–14 (2001).
9) Masuyama, T., Sekine, S. and Nakagawa, H.: Automatic Construction of Japanese KATAKANA Variant List from Large Corpus, *Proc. of the 20th International Conference on Computational Linguistics (COLING)*, pp.1214–1219 (2004).
10) Murawaki, Y. and Kurohashi, S.: Online Acquisition of Japanese Unknown Morphemes using Morphological Constraints, *of the 2008 Conference on Empirical Methods in Natural Language Processing (EMNLP-2008)*, pp.429–437 (2008).
11) Mori, S. and Nagao, M.: Word extraction from corpora and its part-of-speech estimation using distributional analysis, *Proc. of the 11th International Conference on Computational Linguistics (COLING)*, pp. 1119–1122 (1996).
12) 柳原 正，松本一則，池田和史，滝嶋康弘：情報量によるモデル検定を用いた単語境界推定方式の提案，情報処理学会第 190 回自然言語処理研究会論文集，pp.43–47 (2009).
13) Levenshtein, V.I.: Binary Codes Capable of Correcting Deletions, Insertions and Reversals, *Journal of Soviet Physics, Doklady*, pp.707–710 (1966).
14) Kudo, T., Yamamoto, K. and Matsumoto, Y.: Applying Conditional Random Fields to Japanese Morphological Analysis, *Proc. of the 2004 Conference on Empirical Methods in Natural Language Processing (EMNLP-2004)*, pp.230–237 (2004).
15) Nagata, M.: A Stochastic Japanese Morphological Analyzer Using a Forward-DP Backward-A N-Best Search Algorithm, *Proc. of the 15th International Conference on Computational Linguistics (COLING)*, pp.201–207 (1994).