Vol.2016-NL-229 No.15
2016/12/21

単語分かち書き用辞書生成システム NEologd の運用 – 文書分類を例にして –

佐藤 敏紀[1,a)]　橋本 泰一[1,b)]　奥村 学[2,c)]

概要：SNS やニュース記事で発見される新語や未知語の継続的な採録は, 分かち書き用の辞書を作成して更新する際の課題のひとつである. 我々は, 固有名詞や複合語に対する高い網羅性と分かち書き精度を実現し続ける際に, コーパスではなく辞書として言語資源を追加することを選択した. そして, 形態素より長い固有名詞や複合語を単一の見出し語として分かち書きし, 品詞情報や読み仮名などを付与できる辞書を生成するためのシステムを構築した. さらに辞書生成システムを運用して短い周期での更新を長期間継続した. 我々は, 形態素より長い固有名詞や複合語を一語として分かち書きすること, 及び, 辞書生成システムを運用して短い周期で更新することの各効果を, ニュース記事を複数のカテゴリに分類する実験を通じて確認した.

1. はじめに

日本語の言語処理における最も基本的な処理である単語分かち書きには, 現代でも解決できない課題が多数ある. 単語分かち書きのおもな課題は, 分かち書き処理が推定する単語間の境界の精度や, 各単語に付与する品詞情報の精度を改善することである.

単語分かち書き処理のための実装が持つべき重要な機能は新語や未知語に対応することである. とくに大量の Web 文書を実用的な処理時間で扱う場合, ある瞬間に辞書やモデルを更新してから次に更新するまでの間に生まれた新語や, 使用頻度が顕著に上がった未知語は, 学習済みのモデルや事前に用意したルール群を用いて処理することになる.

文中の新語や未知語を形態素解析や分かち書き処理した際の誤り事例はおもに以下の 3 種類に分類できる.

- 1 単語を複数の既知な形態素・単語に分割
- 対象単語に隣接する別の単語の境界を破壊して分割
- 語に誤った品詞情報や読み仮名を付与

分かち書きしたテキストを利用する応用タスクの結果は, 単語分かち書き処理の出力結果に依存する. 最適な分かち書きの粒度はタスクにより変わるが, 未知語や形態素は解析誤りの原因になり, その誤りは後段の処理に影響する.

我々は, 過去に Web サービスにおける文書分類タスクや単語分散表現獲得タスクに業務として取り組んだ際に, 未知の単語や形態素が原因である解析誤りの影響が小さくないことを確認している. また, タスクごとや解析する文書のドメインごとに最適なチャンカーや固有表現抽出器をつくるコストと, それらを保守し続けるコストが高いことも確認している.

そのうえで我々は, 未知の単語・形態素が原因の解析誤りを改善できる単語分かち書き手法を実現したいと考えた. また, 個別の解析誤りへの対処や後処理の実装を繰り返すよりも低コストでその手法を実現し, 可能なら広範なタスクで有向な言語資源を作成したいと考えた.

言語資源の作成によって形態素解析や単語分かち書き処理を改善する方法としては, アノテーション付きコーパスを作成する方法と, 未知語を辞書の語彙として追加する方法の 2 通りが考えられる.

森ら [1] は 1 単語あたりに必要なアノテーション付きコーパスの事例数はおよそ 3 回程度と推定している. その結果を踏まえると, 例えば 100 万種類の異なり語に関する形態素解析や単語分かち書き処理の結果をコーパスを作成する方法で改善するなら, 理想的には 300 万事例という大規模なアノテーション付きコーパスを構築することになる. 仮にその様な大規模コーパスの構築が実現できた場合, そのコーパスは単語分かち書きタスク以外のタスクにおいても極めて有益だといえる.

[1] LINE 株式会社 Data Labs
Data Labs, LINE Corporation
[2] 東京工業大学 科学技術創成研究院 未来産業技術研究所
Laboratory for Future Interdisciplinary Research of Science and Technology, Institute of Innovative Research, Tokyo Institute of Technology
[a)] overlast@linecorp.com
[b)] taiichi.hashimoto@linecorp.com
[c)] oku@pi.titech.ac.jp

しかし我々は, 固有表現や複合名詞に対する高い網羅性と実用的な速度と精度をもっている単語分かち書き処理を短期間に実現する必要があったので, コーパスではなく辞書として言語資源を追加することを選択した.

形態素解析や単語分かち書き処理の結果を改善するうえで, 辞書として言語資源を追加する手法の有効性の高さは知られているのにも関わらず, 企業が有償で提供するサービス以外には頻繁な更新を継続的にはおこなう辞書の提供は行なわれてこなかった.

以上のような背景から本研究では, 既存の形態素解析器の辞書を基にして, 形態素より長い固有名詞や複合語を単一の見出し語とする単語分かち書き用の辞書を作成するためのシステムを実装する. また, 先に述べた新語・未知語の問題に対応する方法として, システムを運用することで辞書に語彙を継続的に追加する手法を提案し, その運用について報告する.

本研究の自然言語処理研究に対するおもな貢献は, 長年, オープンソースソフトウェア (以下, OSS と呼ぶ) な形態素解析辞書が更新されなかったために起きていた新語や未知の固有名詞や複合名詞が原因となる解析誤りを改善したことである.

本稿では, 我々が取り組むタスクの概要 (2 節), システム自体 (3 節), システムの運用と辞書の生成 (4 節) について述べる. さらに提案システムの運用結果の有効性を確かめるための実験をおこなった (5 節). ニュース記事の文書分類の評価セットを作成し (5.1 節), 既存の形態素単位の辞書や異なる日時に更新した辞書の性能の比較をした (5.2 節). その実験により, 新しい語彙が含まれる様に辞書を更新することで, その辞書を使った文書分類システムの性能は既存の形態素解析辞書を使うよりも有意に向上した. また, 辞書を更新しつづけたとしても不利益なことは起きないことを確認した. さらに, 学習データの量を増やすことと辞書を更新することの関係を確認する実験もおこなった (5.3 節). 既存の形態素解析辞書を使用する場合と比べ, 学習データを増やしても性能が急激に悪化せず, 性能を保つか, 劣化する速度を抑えてモデルの頑健性を向上できることが分かった.

2. NEologd で解決するタスクの概要

我々は NEologd というシステムを使った語彙獲得タスクに取り組んでいる. このタスクの目標は, インターネット上で使われた新語や未知の固有表現や複合名詞, サ変接続名詞などの見出し語になる可能性がある表層, 読み仮名, 表層の原型, 品詞情報の 4 つの要素の集合 (以下, 4 つ組と呼ぶ) を収集することである. 表 1 で 4 つ組の各要素について述べる.

NEologd は, Web クローラー群とデータ抽出・結合のためのバッチ処理の組み合わせによって, 収集したデータから自動または半自動的に 4 つ組のリスト (以下, 4 つ組リストと呼ぶ) を生成する. 4 つ組リスト中のエントリ (以下, 単にエントリと呼ぶ) の例を表 2 に示す.

表 1 4 つ組の各要素

要素名	要素の詳細
表層	見出し語の表層形の文字列
読み仮名	表層に付与できる振り仮名のカタカナ表記
原型	表層と対応づく基本形や正式度の高い頻出な表記
品詞情報	ipadic version 2.7.0[2] の IPA 品詞体系の品詞

2.1 4 つ組を収集する際のガイドライン

NEologd で 4 つ組を収集する際の判断基準があるので, 以下でその基準の一部について述べる.

2.1.1 全ての新語と未知語が収集対象ではない

NEologd の自動化された処理の割合を増やすために, 処理対象のテキストデータの質を高めたり, 抽出結果を集計して上位から優先的に判断するなどの工夫は必須である. 語が出現した Web 文書の URL や収集した Web 文書中で語が使用された頻度, 語を使用したユーザーの多さ, 語の出現に季節性や恒常性があるかなどを考慮している. 基準を満たさない語であっても, 流行語は現実の語もインターネット上の語も速やかに登録する. また, 辞書に採録したら面白いと我々が確信した語は積極的に手作業で採録している. もしも, 辞書全体の性能を顕著に低下させる見出し語が見つかった場合は速やかに排除する.

2.1.2 おもに形態素より文字列長が長い単語を収集する

4 つ組のタスクで扱う見出し語の単位に制限はない. そのうえで, 実現したい処理は文を形態素に分割する処理では無いので, 我々は形態素より長い文字列長の固有名詞や複合名詞, サ変接続名詞などを積極的に収集する. 固有名詞や複合名詞はできるだけ長い文字列として扱うことで, 既存の形態素解析処理の後処理として行なっていたチャンキングや固有表現抽出の大半を省略でき, 分かち書きや固有表現に関する品詞情報を付与する際の誤りが減る.

人名のフルネームは Web 上で程度の頻度がある場合には, 姓と名を分割しないでエントリを作る. 早口言葉, ことわざ, 四字熟語, 決まり文句などの, 分かち書きをして形態素列を獲得した場合に, 形態素列を獲得できる以上の利点が無いフレーズは積極的に 1 単語として登録する. 用言や副詞に関しては名詞と比べて新語が生まれる量が少ないので, 年に数回程度, 収集済の Web 文書から未知の用言や副詞を抽出し, まとめて追加する作業をおこなう.

2.1.3 4 つ組の要素の欠損を認めない

4 つ組の要素のどれか一つでも不明な場合は, 最終的なリストに追加しない. リストにエントリを追加する際に 4 つ組の他の要素から明確にならない要素がある場合は, 更にデータを収集するか, 人手で不足している情報を補完する. 例えば, 仮名文字と長音記号のみで構成された表層をカタ

表 2　4 つ組リスト中のエントリの例

表層	読み仮名	原型	品詞情報
東京工業大学	トウキョウコウギョウダイガク	東京工業大学	名詞, 固有名詞, 一般,*,*,*
東京工業大学	トウキョウコウギョウダイガク	東京工業大学	名詞, 固有名詞, 組織,*,*,*
東工大	トウコウダイ	東京工業大学	名詞, 固有名詞, 一般,*,*,*
東工大	トウコウダイ	東京工業大学	名詞, 固有名詞, 組織,*,*,*
MacBook Pro	マックブックプロ	MacBook Pro	名詞, 固有名詞, 一般,*,*,*
東京都渋谷区渋谷	トウキョウトシブヤクシブヤ	東京都渋谷区渋谷	名詞, 固有名詞, 地域, 一般,*,*
東京都渋谷	トウキョウシブヤ	東京都渋谷区渋谷	名詞, 固有名詞, 地域, 一般,*,*
西川仁	ニシカワヒトシ	西川仁	名詞, 固有名詞, 人名, 一般,*,*
2016 年	ニセンジュウロクネン	2016 年	名詞, 固有名詞, 一般,*,*,*
生麦生米生卵	ナマムギナマゴメナマタマゴ	生麦生米生卵	名詞, 固有名詞, 一般,*,*,*

カナに変換して読み仮名にする. 表層を原型として扱えると事前に分かるなら表層を複製して原型にする. クロール先の文書が芸能人のフルネームのリストと分かるなら, 品詞情報は人名にする. 4 つ組の要素のうち読み仮名を必須とする制約は, 収集した 4 つ組の質を一定以上に保つうえで役立つ. そのため, 付与されている読み仮名が不正確な可能性が高いと分かった Web ページや言語資源からは 4 つ組を抽出しない. 表層として略語を使用する際には, その略語のより正式な表記を原型にもつエントリを作成する. その際, その原型を表層に持つエントリを過去に収集していない場合は, 原型を表層に複製して適切な読み仮名を付与したエントリも同時に作成する.

2.1.4　同じ表層のエントリが複数存在して良い

NEologd のタスクは語義曖昧性の解消が目標ではないので, 表層は同じで他の 4 つ組の要素が異なるエントリが存在しても問題無い. 例えば, 地名や組織名, 名字などの様に, 表層から判断できる読み仮名や品詞情報に多義性がある場合, 表層が同じでその他の要素が異なる複数のエントリを生成する. もしも, 生成する複数のエントリに相対的な順序を与えられるなら, 応用時に順序に基づいて異なる判断が可能な様に, 追加の要素として順序を表す数字を設定すると, 将来役に立つ.

2.1.5　より頻出かつ正式な表記を原型にする

我々は 4 つ組の用途を形態素解析に限定していないので, 原型には表層と対応づく基本形や正式度の高い頻出な表記であること以上の制約を設けていない. 4 つ組の品詞情報が名詞系の場合, 我々は原型に入る文字列の正式さを最重要視しない. 例えば「東京工業大学」より「国立大学法人東京工業大学」の方がより正式な表記といえるが, Web 上でより頻出かつ正式さもある「東京工業大学」を原型として採用したい.

3.　NEologd の運用

我々が 2015 年 3 月以降の長期間継続している NEologd[*1] の運用作業は, 語彙の獲得と 4 つ組リスト生成との大きく 2 つに分けられる. 前者の語彙の獲得に関わる処理は大きく以下の 5 種類に分けられる.

(1) 新語や未知語の検出
(2) Web サイトのクロール
(3) 語彙が不足しているドメインに属する用語の網羅
(4) テンプレートによる生成
(5) ホワイトリスト, ブラックリストの管理

これらの語彙の獲得に関わる処理を自動的または半自動的に行ない, その結果から任意のタイミングで 4 つ組リストを生成している. 以下では, 語彙の獲得と 4 つ組リストの生成について述べる.

3.1　新語や未知語の検出

インターネット内外のイベントに連動して Web 上での使用頻度が上がった文字列を検出するために, 新語や未知語の出現を監視する処理を常時実行している. その処理によってニュース記事, Twitter のトレンドワード/ハッシュタグ, 各種検索エンジンの人気キーワード, 放映中の TV 番組名, 掲示板などの時事性がある情報を定期的に獲得し, パターンを用いて固有表現や複合名詞, サ変接続名詞の候補を抽出している. 1 ヶ月以内の出現頻度が上位かつ NEologd のデータベースに未収録な文字列が見つかった場合, それらをリスト化して頻度の降順でソートした後, 人手で確認し 4 つ組化して, 採録・非採録の判断結果と共にデータベースへ登録する. 一定間隔で出現頻度が高い語を検査して取り除くことと, 集計に使用するデータ収集日時が徐々に未来に進むことによって, 新しく検査するリストの上位部分は検査する価値がある文字列の比率が高い. この検査手法はとても素朴な方法だが, 採録の判断を自動化していないので, いまのところ問題が起きていない.

3.2　Web サイトのクロール

Web 上の網羅的かつ定期的に更新される言語資源を記載しているサイトは, クロールする度に前回以降に更新された情報のみを効率よく収集する様に工夫している. クロールする前には Web サイトの利用条件を確認し, robots.txt

[*1] https://github.com/neologd/neologd

が設置されていれば記載されたルールを尊守している. 必要であればコンテンツの公開元に問い合わせをしてクロール・使用許可を頂いている. クロールは常時おこなっており, クロールする範囲は徐々に拡大している. 様々なサイトをクロールするうちに, 単語とその読み仮名の組の正確性が高い Web サイトは以下のどれかの条件を満たしていることが分かった.

- 商業的な理由で正確に読み仮名を付与している
- 複数の人間が編集することで徐々に誤りが減る
- 作者が強い意志で正確性と網羅性を維持している
- Web サイトから特定の個人が利益を得られない

このような条件のどれかを満たす Web サイトをクロールすることで効率のよく 4 つ組を取得できる. 例えば, はてなキーワード [*2](事前に許可を頂いてからクロールしている) や, 日本郵便の郵便番号データダウンロード [*3], 日本全国駅名一覧のコーナー [*4] などが, これらの例に当てはまる. ニコニコ大百科 [*5] のエントリも採用を検討した. しかし, ニコニコ生放送の配信者自身が辞書の項目作ることで PV を得られる構造になっていたため, 大百科全体から効率よく 4 つ組を取得ができずに採用を一旦見送った.

コンテンツの使用許可が明確に得られないサイトから取得した 4 つ組はそのまま採録せずに, 収集した記録を蓄積する. その後, 異なる複数の Web サイトからの収集記録が一定数溜まった 4 つ組については, その読み仮名が既存の言語資源を利用して取得できる場合に採録する.

クロール対象の Web ページから獲得できる情報から品詞情報を詳細に選択できる場合は, その品詞情報を優先する.

3.3 語彙が不足しているドメインに属する用語の網羅

NEologd で獲得したテキストデータ中に出現する単語を監視していると, 専門用語などの特定のドメインに属する単語の大半が採録できていないことが判明する場合がある. その様なドメインに属する語彙の一般性を判断した結果, 専門家以外も使う単語が多く含まれると判断できた場合は, そのドメインに属する単語の網羅的な収集を試みる. NEologd を用いた語彙収集では 4 つ組を作成する必要があるので, 以下の条件のうちどちらかが満たせる情報源が必要である.

- 単語と読み仮名が一組になっているか
- 読み仮名を高精度に獲得するための情報があるか

もしも自動的な用語の採録を行いたいと考えたときは, その様な条件を満たしている Web サイトを複数見つける必要がある. またそれらの Web サイトで使用しているデータが異なる作者や組織によって作成されている必要がある.

そのような複数の Web サイトから獲得した見出し語の表層と読み仮名の組が一致する場合, その表層と読み仮名の組には一定の信頼性があると考え, 自動的に 4 つ組化して採録している.

もしもその様な組を大量に保持している Web サイトの集合を発見できなかった場合は, 生成した 4 つ組を人手または半自動的に精査・修正して採録の可否を判断している. この作業は高コストだが, 将来における必要性が高く見積もれた場合には, 人手が必要な作業も積極的に行っている.

とくに災害時のニュースや情報を処理する際に必要な用語や固有名詞については, 仮に人手の作業が必要な場合でも優先度を上げて作業をしている. 人名や地名については災害発生時よりも前に採録している必要があるので, 定期的に大規模なメンテナンスをして網羅性の向上を試みている. 災害発生後に頻度が上がった単語は, 直近の更新に間に合う様に採録作業を進めているが, 緊急性があると判断した場合には更新頻度を上げて対応している.

以下に不足していると判明したあと網羅的に収集した見出し語の例を挙げる.

3.3.1 例 1. 人名

NEologd の処理において人名は氏名 (フルネーム), 姓 (名字・ラストネーム), 名 (名前・ファーストネーム) の 3 種類に分けている.

人名は Web 文書を解析する上で最も重要だが, 網羅率の維持と向上が難しい. 例えば新語や未知語の出現を監視すると, 我々が知っていても NEologd のデータベースに 4 つ組が含まれていない人名を多く獲得できる. 我々は NEologd の運用当初から, 人名の不足を補うために工藤が公開している人名データ [*6] を利用しているが, それだけでは不十分であった. そのため少なくとも一度は真面目に網羅を試みる必要があった. 我々は有名人の氏名, 大半の名字, 素直に読める名前についての網羅率の向上を試みた.

氏名の網羅を試みる際は, 特定分野に関する氏名リストを獲得して, その氏名リストからルールベースで抽出して, 複数の氏名リストに出現する氏名のみを残すだけで実用上十分な精度が獲得できる.

姓の網羅を試みる際は, はじめに複数の Web サイトから姓リストを獲得する. その後, 表層と読み仮名の組の集合を作成して, 複数の姓リストに出現する表層と読み仮名の組だけ残す. 表層と読み仮名の組に出現した姓リストの数を数え上げ, 同じ表層毎に組を束ねた後, 各組に出現姓リスト数の降順に並べた場合の順序番号を付与する. 表層と読み仮名から生成した各 4 つ組に追加要素としてその順序番号を付与することで, 同じ表層で異なる読み仮名の 4 つ組に異なる重みを与える際のパラメタとして活用する.

名の網羅を試みる際は, はじめに姓と同様に複数の Web

[*2] http://d.hatena.ne.jp/keyword/
[*3] http://www.post.japanpost.jp/zipcode/download.html
[*4] http://www5a.biglobe.ne.jp/ harako/data/station.htm
[*5] http://dic.nicovideo.jp/
[*6] https://twitter.com/taku910/status/47156902429335552

サイトから名リストを獲得して, 表層と読み仮名の組の集合を作成する. その際に, 難読な名前が掲載されたリストも複数獲得して, 難読な名前リストからも表層と読み仮名の組の集合を作成する. その後, 姓と同様にその集合から複数の名リストに出現する表層と読み仮名の組だけ残す. さらに, 難読な名前リストから得た表層と読み仮名の組に一致した組を取り除き, 姓と同様の工程を経て順序番号つきの 4 つ組を獲得する. それだけでは名前を網羅しきれなかったため, 単一の名リストにしか出現しなかった表層と読み仮名の組のうち, 過去に獲得した氏名の 4 つ組と表層と読み仮名がそれぞれ後方一致する組からも 4 つ組を獲得した. 後方一致しなかった場合には, 表層を形態素解析器で処理することで当て字として読み仮名が得られる場合は 4 つ組を獲得した.

この様にすることで有名人の氏名, 大半の名字, 素直に読める名前についての網羅率を向上した. 今後も氏名については随時収集と採録を行ない, 姓と名については年に 1〜2 回程度調査を行ってまとめて採録する.

3.3.2 例 2. Unicode 絵文字

Unicode 絵文字は Unicode の開発を調整するユニコードコンソーシアム [*7] が定義している文字セットの一部で, Unicode 6.0 からは日本の携帯電話で長く使われてきた絵文字が正式に定義されている. 6.0 以降も, 世界中の人々の生活・文化・宗教・社会運動やオリンピックなどの国際的なイベントに連動して新たな絵文字が追加されている. 我々は Unicode 絵文字に関するエントリを Unicode の新バージョンをリリースされる度に追加している. 近年, Unicode 絵文字は世界中の人々が利用しており, とくに SNS で様々な使用事例を確認できる. Unicode 絵文字は文の先頭や末尾における記号としての用途以外に, 文中で一般名詞やサ変接続名詞として使われたり, ハッシュタグの文字列としても使われる. 例えばビールジョッキの絵文字が記号や一般名詞としてのビールと対応する以外に, 文内の出現位置によってはサ変接続名詞としての飲酒と対応する. 我々は各 Unicode 絵文字が記号とハッシュタグ以外に, 文中で名詞やサ変接続名詞の役割で使われる可能性を調査している. その調査にもとづいて 1 つの Unicode 絵文字から最大 4 種類の 4 つ組を作成している. 絵文字の 4 つ組の読み仮名としてどんな読みを付与するかは品詞情報によって変わることがある. また絵文字の使われ方は少しずつ変化し続けているので, 定期的に実際の用法を観察し, より適切な読み仮名が付与される様に更新を続ける必要がある.

3.3.3 例 3. 顔文字

顔文字は文字の組み合わせだけで作られた表情などをもった顔に見える表現のことである. 顔文字は書き手の感情を読み手に対して言語表現だけでは伝えにくい感情表現を補う役割もあり, 文字によるコミュニケーションをより円滑にする目的で広く使われている. 顔文字は Web 上の情報に広く存在しており, 顔文字の存在を確認することは容易である. また, 既存の顔文字の文字列を編集するだけで容易に創作性を発揮できるため, 日々新たなパターンが生み出されている. Web 上には新しい顔文字を配布するサイトが多数あり, また, 顔文字の入力を補助する目的のアプリケーションも様々な環境で多数配布されている. もしも顔文字の一部を削除した場合は残りの文字列が人間が読んでも正しく意味を解釈できないノイズとなる. また顔文字を複数の文字列の系列に分割すると, 顔文字全体によって表現されている繊細な印象が失われる. そのため顔文字はその左端から右端までを 1 つの見出し語として扱う必要がある. 我々が NEologd の 4 つ組として顔文字を収集する際には, 以下の 3 つの問題がある.

- 顔文字の両端はどのようにして検出するか
- 顔文字の読み仮名はどうやって取得するか
- 顔文字の原型は何にしたら良いのか

我々は顔文字の両端を継続的に性能を保って自動推定することが困難だと考えたので, はじめにスマートフォンで入力される顔文字に収集対象を絞った. そして iOS[*8] の標準の日本語入力キーボードと, Android [*9] の Google 日本語入力 [*10] を標準インストールした直後に入力可能な顔文字を全て採録した. その後も, 継続的に Web 上で人気のある顔文字のパターンを追加している.

顔文字の読み仮名を決める作業は難易度が高く, 現状ではカオモジという文字列を仮に与えている. 今後は顔文字を印象毎に分類して付与したラベルにもとづいて読み仮名を決めたり, 顔文字中のテキストを読み仮名として抽出したりする必要があると考えている.

顔文字の原型を決めるためには, 顔文字の原形を抽出する必要がある. 近年 Web 上での使用頻度が高まり派生パターンは UTF-8 で使用できる特殊文字の組み合わせで構成されている. 顔文字の原型を抽出する技術には先行研究があるが, UTF-8 の特殊文字の扱いについては今後の発展が必要になる. 我々は, 顔文字を配布している Web サイトを網羅的にクロールしてデータベースに蓄積し, データベース中での頻度が高い UTF-8 の特殊文字を含む顔文字からも 4 つ組を作成している.

3.4 テンプレートによる 4 つ組生成

収集したデータから得られる 4 つ組の正確さが極めて高いことが分かっている場合には, あらかじめ検討しておいた有益なテンプレートによるエントリ生成もデータが更新されるたびに試みる. 具体的には, 収集したデータやその

[*7] http://unicode.org/

[*8] http://www.apple.com/ios/
[*9] https://www.android.com/
[*10] https://www.google.co.jp/ime/

データから抽出した 4 つ組を生成バターンに入力して, 別の 4 つ組を生成する. 以下では NEologd によるテンプレートを使った 4 つ組生成処理のうち 2 例を挙げる

3.4.1 例 1. 住所文字列

日本郵便の郵便番号データダウンロードから得られる住所の郵便番号データファイル (以下, KENALL.CSV と呼ぶ) は, 郵便番号と住所等を対応させたデータベースで CSV 形式で配布されている. このデータは毎月月末に更新されるので, そのタイミングに合わせてテンプレートによる 4 つ組生成処理を実行する.

KENALL.CSV の形式の詳細については割愛するが, KENALL.CSV 内の住所を表す表層の文字列は都道府県名, 市区町村名, 町域名の 3 つに分かれている. また, 読み仮名も同様に分かれている. 他方, 実用上は都道府県名, 市名, 区町村名, 町域名の 4 つくらいに分かれて欲しい. KENALL.CSV は現実の複雑な住所文字列を記載しているので機械的な処理をしにくい. 大澤が実装した Parse::JapanesePostalCode [*11] という Perl モジュールは, 実用上困らない程度の厳密さで都道府県名, 市名, 区町村名, 町域名を獲得できる様に KENALL.CSV を加工できる. KENALL.CSV 中の, 神奈川県横浜市緑区長津田町いう住所に関する行を例に挙げると, この住所はそのまま NEologd の 4 つ組に変換できる. さらに Parse::JapanesePostalCode で神奈川県/横浜市/緑区/長津田町と区切り, 町域名以外の一部を削除した文字列をあらかじめ決めたテンプレートで生成し, 自動的に採録できるか判断をしている. 以下に生成した結果採録可能だった表層を示す.

- 神奈川県横浜市緑区長津田町
- 神奈川県横浜市長津田町
- 神奈川県緑区長津田町
- 神奈川県長津田町
- 横浜市緑区長津田町
- 横浜市長津田
- 緑区長津田町

都道府県名, 市名, 区町村名, 町域名のうち一部のレベルをマスクした文字列群を生成するためのテンプレートを KENALL.CSV から得られる全ての住所に適用し, 出現頻度が 1 な文字列だけを 4 つ組の表層として扱う. その表層を生成する際の基になった住所文字列は表層と一意に対応する原型として扱える. 例えば, 今回は上記の表層すべての原型は神奈川県横浜市緑区長津田町になる.

3.4.2 例 2. 時間表現と数値表現

NEologd は固有名詞の時間表現や数値表現 [3][4][5][6] の 4 つ組を, 事前に定義したテンプレートを用いて生成している. 3.1 節で述べた新語検出の処理の結果中に, 複数の形態素に分割されてしまう高頻度な時間表現や数値表現を検出した際に, その時間表現や数値表現を網羅するテンプレートを追加する. そして, NEologd の見出し語データをまとめ上げるタイミングで時間表現と数値表現の 4 つ組を生成し直している. 表 3 に時間表現と数値表現を生成するテンプレートの例を示す.

例えば, 第 4 四半期, という時間表現を検出したとする. その場合は可変な数字に対して接頭辞”第”と接尾辞”四半期”を結合するが, その際に可変な数字 1 から 4 まで 1 づつ増え, 原型は生成した表層と同じ』というルールをテンプレートとして書く. このルールによって第 1 四半期から第 4 四半期までの 4 つのエントリが生成される. 日付表現や数量表現は際限なく生成できるため, 我々の想定する応用方法における実用性と生成するエントリ数のバランスを考えて, テンプレートの追加・削除や各テンプレートのパラメタを調整している. 表 3 には書いていないが, 実際にはテンプレートに変数部分や接尾辞の読み仮名の音便に関する設定も書いている. また, 1 年目を表す元年の様に変数部分の表記や読み仮名を生成するために, 特殊な規則や知識が必要な場合は積極的に人手でルールを足す.

3.4.3 ホワイトリスト, ブラックリストの管理

個別の見出し語の採録・非採録を決定するためのルールは少ないことが望ましいが, 実用上はすばやく問題を解決する必要がある場合もあり, 4 つ組リストを生成する際に参照するホワイトリストとブラックリストを用意している. ホワイトリストに記載された 4 つ組は一部のフィルタリングルールを回避して, 相当大きな問題が無い限り 4 つ組リストに採録される. ブラックリストに記載された 4 つ組は, 様々な処理の区切れ目でチェックをおこない, 新しい処理を追加した際に相当大きなミスが無い限り非採録になる. やむを得ずブラックリストに足した見出し語の例を挙げると, お笑い芸人の『ですよ。』という芸名は, 末尾が『ですよ。』で終わる全ての文の解析結果に悪影響を与えるので足した. ホワイトリストに足した見出し語の例を挙げると, ある時に『しらたき』という食べ物を指す一般名詞『白滝』のひらがな表記を表層とする 4 つ組を採録できていないことに気がついた. 我々は『白滝』だけに特化しない手法で『白滝』をひらがな化した表層の抽出を複数試みたが, 同時に『しらたき』以外のノイズとなる表層も多く出力されてしまい, そのノイズを精度良く抑制または削除できなかった. 『しらたき』という表記が使用される頻度とノイズを除去するコスト, ホワイトリストに足すコストを鑑みて, ホワイトリストに足した. どちらのリストも登録されている 4 つ組は極めて少ないので, 定期的に見直して不要になった 4 つ組は消す予定である.

4. NEologd で収集した 4 つ組データの応用

3 章で紹介した NEologd とその他に人手で蓄積した 4 つ組リストからは様々な言語資源を作れるが, 以下ではおも

[*11] https://github.com/yappo/p5-Parse-JapanesePostalCode

表 3 時間表現と数値表現を生成するテンプレートの例

		表層の接頭辞		表層の変数			表層の接尾辞		原型		
生成する見出し語	固有表現タイプ	表記	読み仮名	初期値	差分	最大値	表記	読み仮名	接頭辞の表記	変数の初期値	接尾辞の表記
2000 年 ～ 2050 年	時間表現	-	-	2000	1	2050	年	ネン	-	-	-
平成 2 年 ～ 平成 28 年	時間表現	平成	ヘイセイ	2	1	28	年	ネン	-	1990	年
第 1 四半期 ～ 第 4 四半期	数値表現	第	ダイ	1	1	4	四半期	シハンキ	-	-	-
0 歳 ～ 125 歳	数値表現	-	-	0	1	125	歳	サイ	-	-	-
0 才 ～ 125 才	数値表現	-	-	0	1	125	才	サイ	-	0	歳

に単語分かち書き用辞書の生成について述べる.

4.1 単語分かち書き用辞書 mecab-ipadic-NEologd

我々は mecab-ipadic-NEologd[*12] という MeCab[7] の辞書として使用できる単語分かち書き用辞書を作成している. この辞書は工藤さんが公開している MeCab 用の IPA 辞書 (以下 IPADIC と呼ぶ) をベースにしている. まずはじめに『国土交通省は 2001 年に設置されました。』という文を例として, 形態素解析器 MeCab の辞書として mecab-ipadic-NEologd と IPADIC と UniDic [*13] を使った場合の分かち書き結果と, KyTea 0.4.7[8] による分かち書き結果, Juman++ 1.01[9] による分かち書き結果を表 4 に示す.

解析器と辞書による分かち書きの結果に特色があるが, mecab-ipadic-NEologd の分かち書き結果以外への言及は割愛する. mecab-ipadic-NEologd は NEologd で生成した 4 つ組リストを用いて, MeCab で使用可能なフォーマットの CSV ファイルに変換している. その際に 4 つ組リストに含まれ無い『左・右文脈 ID』『形態素生起コスト』『発音』を獲得する処理を実行している. 表 4 から明らかな様に, NEologd の 4 つ組リストを使っているので mecab-ipadic-NEologd は形態素に分かち書きするための辞書ではなく, 形態素よりも長い固有名詞や複合名詞に分かち書きするための辞書になっている. また, mecab-ipadic-NEologd は『固有名詞や複合名詞を形態素に分割しない』という観点に反しない場合は, mecab-ipadic-NEologd と IPADIC の分割結果がなるべく一致する様に調整している. また, IPADIC のみで上記の観点から見て正しく解析できている事例に悪影響を与える見出し語は mecab-ipadic-NEologd から取り除いている.

4.2 mecab-ipadic-NEologd の長所と限界について

mecab-ipadic-NEologd に可能なことと不可能なことを分かりやすく示すため, 『任天堂のミニファミコンの販売数は発売日から 4 日間で 26.3 万台に達した.』という文を MeCab で解析した際の辞書による解析結果の違いを, IPADIC を使った場合 (表 5) と, 2016 年 11 月 3 日に更新された mecab-ipadic-NEologd を使った場合 (表 6) とに分けて示す.

表 5 と表 6 を比べると, IPADIC に採録されている『任天堂』以外に, mecab-ipadic-NEologd は『ミニファミコン』, 『4 日間』などが取得できている. これは NEologd が固有名詞や日付表現を収集・生成しているからである. また, mecab-ipadic-NEologd は『ミニファミコン』の正式な表記とその略語を共に採録できている. 『発売日』が一般名か固有名詞かは判断が難しいが, IPADIC の一般名詞と同様の品詞情報を与えるためには, 一般名詞とは何か, や, IPADIC に採録されていない一般名詞は何かという問題について考える必要があるので判断を据え置く. 『販売 / 数』は NEologd の観点では 1 つの見出し語になりそうだが, 2016 年 11 月 3 日の時点では未再録である. しかし, NEologd の新語・未知語検出処理は監視範囲内での出現頻度が高い固有表現や複合名詞を検出できるため, 近いうちに採録されると思われる.

この様に既存の形態素解析や固有表現抽出技術の課題として挙げられてきた語彙が足りない問題や, 新語や未知語に対処できない問題については, mecab-ipadic-NEologd によって大幅に軽減できているし, 今後も改善し続ける.

他方, mecab-ipadic-NEologd が解決できない問題も大きく 2 つある. ひとつは, 専門家しか使わない固有表現や複合名詞の様に, 採録の条件が揃わない可能性が高い語は, 我々が網羅的な登録作業を行うまで 4 つ組が採録されないという問題である. 不足している単語を広い範囲で常に検出し, それを足し続けることは困難である. もうひとつは, 『26 / . / 3 / 万 / 台』のような数値表現に代表される, 事前に見出し語を大量に生成しなければ正しく分かち書きできない単語は, NEologd の 4 つ組リストに含めることが難しいという問題である. 典型的な例としては世界中の通貨ごとの金額や, 製品の型番, 電話番号などが挙げられる.

辞書として言語資源を追加する手法では効率よく対処できない単語を含む文を, 正しく解析・分かち書きするためには, 解析器自体の機能拡張や固有表現抽出技術の精緻化が必要である.

4.3 IPADIC の改善

mecab-ipadic-NEologd を改善するうちに, IPADIC の不具合によって起きる分かち書きの誤りや, 読み仮名の振り間違えを見つけることがある. 分かりやすい例としては『日本酒』が分割されてしまう問題や, 『人民元』の読み仮名が間違っている問題などが挙げられる. これらの問題に対処するため, mecab-ipadic-NEologd はインストール時に, ベースとなる IPADIC に独自のパッチ (変更すべき箇所をまとめたファイル) による訂正処理を適用して, その後にインストールしている.

IPADIC の不具合によって起きる分かち書きの誤りを訂

[*12] https://github.com/neologd/mecab-ipadic-neologd
[*13] http://pj.ninjal.ac.jp/corpus_center/UniDic/

表 4 辞書による分かち書き結果の違い

解析器と辞書の名前	分かち書きの結果
MeCab & mecab-ipadic-NEologd	国土交通省 / は / 2001 年 / に / 設置 / さ / れ / まし / た / 。
MeCab & IPADIC	国土 / 交通省 / は / 2001 / 年 / に / 設置 /さ / れ / まし / た / 。
MeCab & UniDic	国土 / 交通 / 省 / は / 2 / 0 / 0 / 1 / 年 / に / 設置 / さ / れ / まし / た / 。
KyTea	国土 / 交通 / 省 / は / 2001 / 年 / に / 設置 / さ / れ / ま / し / た / 。
Juman++	国土 / 交通 / 省 / は / 2001 / 年 / に / 設置 /さ / れ / ました / 。

表 5 MeCab & IPADIC による解析結果の例

表層	品詞情報	原型	読み仮名	発音情報
任天堂	名詞, 固有名詞, 組織,*,*,*	任天堂	ニンテンドウ	ニンテンドー
の	助詞, 連体化,*,*,*,*	の	ノ	ノ
ミニ	名詞, 一般,*,*,*,*	ミニ	ミニ	ミニ
ファミコン	名詞, 一般,*,*,*,*	ファミコン	ファミコン	ファミコン
の	助詞, 連体化,*,*,*,*	の	ノ	ノ
販売	名詞, サ変接続,*,*,*,*	販売	ハンバイ	ハンバイ
数	名詞, 接尾, 一般,*,*,*	数	スウ	スー
は	助詞, 係助詞,*,*,*,*	は	ハ	ワ
発売	名詞, サ変接続,*,*,*,*	発売	ハツバイ	ハツバイ
日	名詞, 接尾, 一般,*,*,*	日	ビ	ビ
から	助詞, 格助詞, 一般,*,*,*	から	カラ	カラ
4	名詞, 数,*,*,*,*			
日間	名詞, 接尾, 助数詞,*,*,*	日間	ニチカン	ニチカン
で	助詞, 格助詞, 一般,*,*,*	で	デ	デ
26	名詞, 数,*,*,*,*			
.	名詞, サ変接続,*,*,*,*			
3	名詞, 数,*,*,*,*			
万	名詞, 数,*,*,*,*	万	マン	マン
台	名詞, 接尾, 助数詞,*,*,*	台	ダイ	ダイ
に	助詞, 格助詞, 一般,*,*,*	に	ニ	ニ
達し	動詞, 自立,*,*, 五段・サ行, 連用形	達す	タッシ	タッシ
た	助動詞,*,*,*, 特殊・タ, 基本形	た	タ	タ
。	記号, 句点,*,*,*,*	。	。	。

正する際には, はじめに分かち書きの誤りが起きる見出し語をなるべく網羅的に見つける. その方が訂正すべき見出し語が 1000 語を超えたあたりからは, 個別に誤りを訂正するよりも最終的には効率が良い. 次に, 分かち書きの誤りが起きる語の形態素生起コストを, 分かち書きの誤りが減るように下げる. さらにコストを調整した見出し語を含めた辞書を再構築し, 再び分かち書きの誤りが起きる見出し語をなるべく網羅的に見つける. この繰り返しを, 分かち書きの誤りが起きる見出し語の数が収束するまで行い, その結果に基いてパッチを作成する. mecab-ipadic-NEologd は現状では IPADIC の名詞系のエントリのうち約 2.6%のコスト調整をおこなうパッチを用いている.

IPADIC の読み仮名の間違えは 1 件ずつ確認していくのではなく, 他の言語資源と比較して, その結果にもとづいて論述に検出すると効率が良い. この作業は人手が必要なので開発イベントなどを開催し, 集団で修正すると効率が良いと考えている. 過去に我々は実際に合宿形式のイベントで読み仮名の間違えに関する分析と訂正をおこなった.

4.4 mecab-ipadic-NEologd の更新作業

我々が mecab-ipadic-NEologd を週 2 回 (現在は毎週月・木曜日) の頻度で更新する際に行っている作業を以下に挙げる.

- NEologd で生成した 4 つ組リストの取得
- 4 つ組リストから MeCab 用の CSV ファイルを生成
- パッケージング
- GitHub にリリース / 広報活動

上記の作業のうち, 4 つ組リストを取得して CSV ファイルを生成するところまでは自動化しているが, リリースできる形態に仕上げるパッケージングの作業は人手を割いている. 現状では mecab-ipadic-NEologd をリリースする前に CSV ファイルの値の範囲のチェックや, 正常にインストールできるかどうか, インストールした辞書のベンチマーク上の性能, 実際の使用感などを調べてからリリースしている.

リリース直前に世の中で大きなニュースがあった場合は, そのニュースに関連する見出し語が採録済みかを調べ, 当日に採録すべき語をみつけた場合は, NEologd への 4 つ組を登録以降やり直す.

そのほかに GitHub からの Issue や PullRequest への対応, Twitter やはてなブックマーク, Google の Web 検索, Qiita などの検索結果からのリクエストや不具合情報の検出, 具体例のヒアリングや, 質問に対する回答などを随時行っている. Web 上のユーザのリクエストはソフトウェアの不備を改善するうえで貴重なので, 開発者や研究者は努めて情報を収集するべきだと考えている.

4.5 mecab-ipadic-NEologd の改善

mecab-ipadic-NEologd を作成するにあたり, NEologd による 4 つ組リストの収集だけでは足りない見出し語があったので足した. 具体的には用言と副詞, 感動詞, 一般名詞・固有名詞・サ変接続名詞の表記揺れ, 形容詞の崩れ表記語などである.

現状では用言は形容詞と名詞の形容動詞語幹について, IPADIC に採録されていない見出し語を網羅的に採録し終わっている. その際に SNS などで頻出な長音記号の多用や母音仮名文字の連続などにも対応した. 動詞については近日中に採録予定で 2016 年 11 月の時点では作業を進めて

表 6 MeCab & mecab-ipadic-NEologd-20161103 による解析結果の例

表層	品詞情報	原型	読み仮名	発音情報
任天堂	名詞, 固有名詞, 組織,*,*,*	任天堂	ニンテンドウ	ニンテンドー
の	助詞, 連体化,*,*,*,*	の	ノ	ノ
ミニファミコン	名詞, 固有名詞, 一般,*,*,*	ニンテンドークラシックミニファミリーコンピュータ	ミニファミコン	ミニファミコン
の	助詞, 連体化,*,*,*,*	の	ノ	ノ
販売	名詞, サ変接続,*,*,*,*	販売	ハンバイ	ハンバイ
数	名詞, 接尾, 一般,*,*,*	数	スウ	スー
は	助詞, 係助詞,*,*,*,*	は	ハ	ワ
発売日	名詞, 固有名詞, 一般,*,*,*	発売日	ハツバイビ	ハツバイビ
から	助詞, 格助詞, 一般,*,*,*	から	カラ	カラ
4 日間	名詞, 固有名詞, 一般,*,*,*	4 日間	ヨッカカン	ヨッカカン
で	助詞, 格助詞, 一般,*,*,*	で	デ	デ
26	名詞, 数,*,*,*,*			
.	記号, 一般,*,*,*,*			
3	名詞, 数,*,*,*,*			
万	名詞, 数,*,*,*,*	万	マン	マン
台	名詞, 接尾, 助数詞,*,*,*	台	ダイ	ダイ
に	助詞, 格助詞, 一般,*,*,*	に	ニ	ニ
達し	動詞, 自立,*,*, 五段・サ行, 連用形	達す	タッシ	タッシ
た	助動詞,*,*,*, 特殊・タ, 基本形	た	タ	タ
。	記号, 句点,*,*,*,*	。	。	。

いる.

感動詞については Web 上で頻出する感動詞を追加する仕組みを作り, 1 年に数回程度の採録をしている.

一般名詞・固有名詞・サ変接続名詞の表記揺れを吸収するための見出し語は, 形態素解析結果の N-Best 解を再帰的に求めて形態素の木をつくり, ルールベースで枝刈りと経路の列挙を行うことで生成している.

SNS 上に現れやすい崩れ表記語は今後網羅的な解決を試みたいが, はじめに形容詞の崩れ表記語をパターンで生成した.

4.6 カラム拡張データについて

mecab-ipadic-NEologd は形態素よりも長い固有名詞や複合名詞を一語とする分かち書きをおこなう目的で開発しているが, 用途によっては例えば, 固有名詞や複合名詞を IPADIC や UniDic で分かち書きした時にどの位置で分割されるか, の様な MeCab を使った形態素解析の枠組みでは単純に得られない情報が欲しい場合がある.

その様な形態素解析結果以上の結果を獲得するための仕組みとして, mecab-ipadic-NEologd はカラム拡張データと呼ぶデータを利用できる.

このデータは決められたフォーマットで配置された, 表層と文脈 ID をキーとした値のリストである. この表層と文脈 ID は, mecab-ipadic-NEologd のエントリのいずれかと対応する様にする. インストール時に mecab-ipadic-NEologd は配置されたリストと, mecab-ipadic-NEologd の見出し語リストを, 表層と文脈 ID をキーとして結合する. 結合が上手くいった場合は, リストの値を対応する mecab-ipadic-NEologd のエントリの末尾に付与する. この様にすることで, MeCab を使った形態素解析結果の末尾のカラムから, 任意の情報を得られる様になる.

4.7 適用したオープンソースライセンス

mecab-ipadic-NEologd は OSS として GitHub 上で公開している. オープンソースライセンスは Apache License, Version 2.0[*14] のみを適用している. このライセンスを適用している理由は, mecab-ipadic-NEologd を使用する方や, mecab-ipadic-NEologd の解析結果を使用する方が, 自分の開発物のライセンスに関する無用な検討をする時間を削減したいと考えたからである.

mecab-ipadic-NEologd に Apache License, Version 2.0 のみを適用できるように, 辞書構築やインストールは様々な工夫している. Web 上の言語資源には様々なライセンスが付与されているが, 最終的に Apache License, Version 2.0 以外のライセンスを適用できなくなる可能性がある言語資源は, どれほど有益でも mecab-ipadic-NEologd に取り込んでいない. また, 我々が開発中に IPADIC に適用されたライセンスについて考慮する必要が無いように, mecab-ipadoc-NEologd はインストールの直前まで IPADIC のパッケージをダウンロードしない.

mecab-ipadoc-NEologd は我々にとっても必須で基礎的な言語資源であるため, 今後も現状と同様の頒布体制を保ちたいと考えており, 自由さを大切にしたいと考えている.

5. 評価実験

我々は 4.1 節で作成手法を述べた mecab-ipadic-NEologd を現実の Web サービスの機能に適用した場合の効果を測定したいと考えた. 1 章で述べた様に, 我々は過去に文書分類タスクにおいて, 未知の単語や形態素が原因である解析誤りの影響があることを確認している. そこで, 今回はその影響の大きさと mecab-ipadic-NEologd を作成したことによる改善の幅を調べるため, ニュース記事のカテゴリ分類において分かち書きに使用する辞書の違いが, 実験結果に与える影響を調べるための実験を行った.

実験の結果を踏まえて, 固有名詞や複合語を語の単位にすることの長所と短所, および, 辞書を定期的に更新することによる利益や, 辞書を更新するのではなく学習データを増加することによる影響について議論したいと考えた.

以下におこなった実験の詳細と結果を述べる.

5.1 実験で使用するデータセットの構築

ニュース記事のカテゴリ分類における実験をおこなうため, はじめにデータセットを構築した. 様々な Web 上の

[*14] https://www.apache.org/licenses/LICENSE-2.0

ニュースサイトやそのサイトに掲載された記事に設定されたカテゴリの階層構造, 付与されたカテゴリラベルの質などを考慮した結果, 今回は Yahoo!ニュース [*15] から収集した複数日分のニュース記事でデータセット (以後, ニュースデータセットと呼ぶ) を構築した. そのニュースデータセットの詳細を表 7 に示す.

表 7 ニュースデータセットの詳細

収集したサイト	Yahoo!ニュース
収集手法	新着ニュース一覧から定期的に収集
クロールした期間	2016/05/21(土)〜2016/10/28(金)
クロールした記事数	計 539,524 記事

ニュースデータセットの各記事には 8 種類の大粒度と 81 種類の小粒度の 2 つのカテゴリラベルがそれぞれ 1 つずつ付与されている. このカテゴリラベルはニュース記事の収集時に, 各記事が掲載されていた Web ページの HTML ファイルから獲得したものである. 以降ではニュースデータセットの各記事に大カテゴリのラベルを付与した状態を大カテゴリ記事セットと呼び, 同様に小カテゴリのラベルを付与した状態を小カテゴリ記事セットと呼ぶ.

5.1.1 評価データセットの作成

評価にはニュースデータセットから表 8 に示した範囲だけを取り出したものを評価データセットとして扱い, すべての実験結果に使用した.

表 8 評価データセットのデータ収集期間と記事数

セット名	使用するデータの収集期間	記事数
評価データセット	2016-10-21(金) 〜 2016-10-28(金)	28,111

5.1.2 学習データセットの作成

学者データは新たにニュースデータセットから期間を 1 ヶ月ずつずらして抽出して作成した. 表 9 に作成したデータセット (以下, 学習データセットと呼ぶ) の詳細を示す.

表 9 学習データセットのデータ収集期間と記事数

セット名	使用するデータの収集期間	日数	記事数
1 ヶ月分	2016-09-21(水) 〜 2016-10-20(木)	29 日	101,203
2 ヶ月分	2016-08-21(日) 〜 2016-10-20(木)	59 日	204,952
3 ヶ月分	2016-07-21(木) 〜 2016-10-20(木)	90 日	305,272
4 ヶ月分	2016-06-21(火) 〜 2016-10-20(木)	120 日	407,242
5 ヶ月分	2016-05-21(土) 〜 2016-10-20(木)	151 日	511,413

学習データセット中の各記事には大と小のカテゴリラベルが付与されているので, 各セットを大カテゴリ記事セットとしても, 小カテゴリ記事セットとしても使える.

[*15] http://news.yahoo.co.jp/

5.1.3 学習データセットと評価データセットの詳細

ニュースデータセットのカテゴリごとの記事数を確認するため, 評価データセットと学習データセット 1 ヶ月分の各記事に付与されているカテゴリラベルを粒度別に集計した結果を表 10 に示す.

表 10 に示した通り各カテゴリに属する記事数は異なる. 大カテゴリ記事セットの記事の比率は 1 年を通してそれほど変化しない. 他方, 小カテゴリ記事セットの地域系の記事は, 各地域での行事や事件, 地域振興などによって記事が増減する. 例えば, ニュースデータセット中の 2016-09-21(水) 〜 2016-10-28(金) の期間に収集した記事のうち,『地域-沖縄』のラベルが付与されていた記事は 1800 件含まれていたが, ニュースデータセット外の 2016-03-25(金)〜2016-04-29(金) の期間には 1011 件しか含まれていなかった. この時期の沖縄には台風や米軍基地問題, 機動隊員の発言など, 国民が関心を持つ大切なできごとが多く起きていた. 他方, ニュースデータセット外の 2016-03-25(金)〜2016-04-29(金) に収集した記事に『地域-山形』が 517 件付与されていたが, ニュースデータセット中の 2016-09-21(水) 〜 2016-10-28(金) の期間には 18 件しか含まれていない. 3 月末頃は山形県で誘拐監禁事件などがあり頻繁にニュースで取り上げられていた. この様にニュースデータセットの期間を区切って部分的なニュースデータセットを複数作った場合に, 各セットごとの記事件数を揃えたとしても, 各カテゴリに含まれる記事件数が変わる性質がある.

5.2 大カテゴリ記事セットによる辞書の比較実験

表 10 に示した 2016-09-21(水)〜2016-10-20(木) の 101,203 記事を学習データとして, 大カテゴリ記事セットを使ったカテゴリ分類実験をおこなう.

はじめに, あらかじめ典型的な文字列正規化処理をした各ニュース記事からタイトルと本文を抽出し, それぞれを形態素解析エンジン MeCab を使って分かち書きした. その際に表 11 に示した 5 種類の辞書を使用した.

分かち書き処理と同時に獲得できる単語の原型の文字列がその単語の表層の文字列と異なる場合, 原型の文字列に置換した. その後, タイトルと本文の区別をせずに単語の頻度を集計した. 事前に学習時に使用する単語を品詞情報で限定することも試した結果, UniDic のみ名詞だけを使用した場合にすべての単語を使った場合より性能が高かった. しかし, それでも他の辞書を使った場合の性能よりも低くかった. 今回実験した範囲では UniDic 以外の辞書では, すべての単語を使った場合に総合的な性能がもっとも良かったため名詞のみを使った場合の結果は割愛する.

今回は学習器に LIBLINEAR[*16] を使用する。学習データの各記事の頻度を学習データ内における TFIDF 値に変

[*16] http://www.csie.ntu.edu.tw/ cjlin/liblinear/

表 10 ニュースデータセットの各記事のカテゴリラベルを粒度別に集計した結果

大カテゴリ記事セット			小カテゴリ記事セット		
ラベル	学習データの記事数	モデルデータの記事数	ラベル	学習データの記事数	モデルデータの記事数
国内	10550	3131	国内-政治	7977	2388
			国内-社会	2518	719
			国内-人	55	24
国際	8148	2215	国際-国際総合	2932	811
			国際-中国・台湾	749	198
			国際-韓国・北朝鮮	2211	611
			国際-アジア・オセアニア	545	146
			国際-北米	863	226
			国際-中南米	189	38
			国際-ヨーロッパ	431	125
			国際-中東・アフリカ	228	60
経済	10002	2954	経済-経済総合	6913	2005
			経済-市況	875	258
			経済-株式	462	203
			経済-産業	1752	488
エンタメ	24567	6936	エンタメ-エンタメ総合	12656	3498
			エンタメ-音楽	5775	1604
			エンタメ-映画	3099	919
			エンタメ-ゲーム	2068	679
			エンタメ-アジア・韓流	969	236
スポーツ	28583	7330	スポーツ-スポーツ総合	6220	1656
			スポーツ-野球	9465	2407
			スポーツ-サッカー	6274	1569
			スポーツ-モータースポーツ	1023	286
			スポーツ-競馬	2601	676
			スポーツ-ゴルフ	1934	511
			スポーツ-格闘技	1066	225
IT・科学	4585	1466	IT・科学-IT 総合	3648	1219
			IT・科学-科学	333	80
			IT・科学-製品	604	16
ライフ	2055	552	ライフ-ライフ総合	1594	417
			ライフ-ヘルス	155	45
			ライフ-環境	38	12
			ライフ-文化・アート	268	78
地域	12713	3527	地域-北海道	340	98
			地域-青森	523	138
			地域-岩手	72	18
			地域-宮城	97	33
			地域-秋田	48	17
			地域-山形	18	0
			地域-福島	936	251
			地域-東京	669	192
			地域-神奈川	892	278
			地域-埼玉	519	155
			地域-千葉	238	69
			地域-茨城	86	21
			地域-栃木	94	27
			地域-群馬	185	42
			地域-山梨	53	19
			地域-新潟	107	34
			地域-長野	104	34
			地域-富山	404	103
			地域-石川	490	132
			地域-福井	162	43
			地域-愛知	41	7
			地域-岐阜	179	48
			地域-静岡	1278	353
			地域-三重	42	10
			地域-大阪	445	127
			地域-兵庫	629	149
			地域-京都	570	157
			地域-滋賀	217	52
			地域-奈良	88	21
			地域-和歌山	292	82
			地域-鳥取	13	4
			地域-島根	16	8
			地域-岡山	138	39
			地域-広島	71	20
			地域-山口	33	7
			地域-徳島	17	7
			地域-香川	33	7
			地域-愛媛	404	124
			地域-高知	9	2
			地域-福岡	145	59
			地域-佐賀	167	40
			地域-長崎	156	41
			地域-熊本	23	8
			地域-大分	33	9
			地域-宮崎	182	54
			地域-鹿児島	28	15
			地域-沖縄	1427	373

表 11 分かち書きに使用した形態素解析辞書

名前	詳細
IPADIC v2.7.0	配布されている IPADIC をインストールした
UniDic v2.1.2	配布されている UniDic をインストールした
NEologd 20160919	学習データ収集期間前の 2016/09/19 に更新
NEologd 20161021	学習データ収集期間後の 2016/10/21 に更新
NEologd 20161103	評価データ収集期間後の 2016/11/03 に更新

換した後, その値を LIBSVM[*17] の svm-scale コマンドで 0 から 1 の値にスケーリングした. テストデータは各記事の頻度をテストデータ内における TFIDF 値に変換した後で, 学習データをスケーリングした際に保存されたスケール尺度を使ってスケーリングした. その結果として得られた特徴ベクトルと記事に付与してあるカテゴリラベルを組にして, liblinear 形式の学習データと評価データを生成した.

実験の準備段階では, liblinear の学習時のソルバーと c パラメタとバイアス項の設定をニュースデータセットを使用して探索した. その結果, 『s = 5 c = 0.8 B = -1』という設定が比較対象である UniDic に有利だったので, その設定を使用して他のパラメタの組み合わせの結果を割愛する. 上記の設定を使用しておこなった大カテゴリ記事セットの結果を表 5.2 に示す.

国内カテゴリにおける適合率は, 最新の NEologd 20161103 を使 用した場合に最も高い値を得られて, 再

[*17] http://www.csie.ntu.edu.tw/ cjlin/libsvm/

表 12 大カテゴリ記事セット (30 日分で学習) における実験結果

辞書名	IPADIC v2.7.0			UniDic v2.1.2		
クラス名	適合率 (TP/TP+FP)	再現率 (TP/TP+FN)	F 値	適合率 (TP/TP+FP)	再現率 (TP/TP+FN)	F 値
国内	0.772(2388/3095)	0.763(2388/3131)	0.767	0.669(2300/3439)	0.735(2300/3131)	0.700
国際	0.908(1934/2131)	0.873(1934/2215)	0.890	0.893(1705/1909)	0.770(1705/2215)	0.827
経済	0.838(2443/2916)	0.827(2443/2954)	0.832	0.878(1975/2250)	0.669(1975/2954)	0.759
エンタメ	0.937(6626/7071)	0.955(6626/6936)	0.946	0.865(6598/7627)	0.951(6598/6936)	0.906
スポーツ	0.957(7173/7492)	0.979(7173/7330)	0.968	0.892(7209/8080)	0.983(7209/7330)	0.935
IT・科学	0.857(1116/1302)	0.761(1116/1466)	0.806	0.880(850/966)	0.580(850/1466)	0.699
ライフ	0.693(251/362)	0.455(251/552)	0.549	0.858(127/148)	0.230(127/552)	0.363
地域	0.804(3008/3742)	0.853(3008/3527)	0.828	0.754(2785/3692)	0.790(2785/3527)	0.772

辞書名	NEologd 20160919			NEologd 20161021		
クラス名	適合率 (TP/TP+FP)	再現率 (TP/TP+FN)	F 値	適合率 (TP/TP+FP)	再現率 (TP/TP+FN)	F 値
国内	0.774(2354/3041)	0.752(2354/3131)	0.763	0.781(2366/3030)	0.756(2366/3131)	0.768
国際	0.904(1932/2138)	0.872(1932/2215)	0.888	0.903(1937/2145)	0.874(1937/2215)	0.888
経済	0.835(2481/2970)	0.840(2481/2954)	0.837	0.834(2464/2955)	0.834(2464/2954)	0.834
エンタメ	0.934(6619/7084)	0.954(6619/6936)	0.944	0.937(6621/7065)	0.955(6621/6936)	0.946
スポーツ	0.960(7155/7453)	0.976(7155/7330)	0.968	0.959(7168/7476)	0.978(7168/7330)	0.968
IT・科学	0.859(1140/1327)	0.778(1140/1466)	0.816	0.851(1142/1342)	0.779(1142/1466)	0.813
ライフ	0.716(262/366)	0.475(262/552)	0.571	0.703(260/370)	0.471(260/552)	0.564
地域	0.807(3013/3732)	0.854(3013/3527)	0.830	0.808(3014/3728)	0.855(3014/3527)	0.831

辞書名	NEologd 20161103		
クラス名	適合率 (TP/TP+FP)	再現率 (TP/TP+FN)	F 値
国内	0.778(2342/3011)	0.748(2342/3131)	0.763
国際	0.906(1940/2141)	0.876(1940/2215)	0.891
経済	0.835(2468/2955)	0.835(2468/2954)	0.835
エンタメ	0.939(6636/7064)	0.957(6636/6936)	0.948
スポーツ	0.959(7164/7473)	0.977(7164/7330)	0.968
IT・科学	0.856(1139/1331)	0.777(1139/1466)	0.815
ライフ	0.692(267/386)	0.484(267/552)	0.570
地域	0.802(3009/3750)	0.853(3009/3527)	0.827

現率は IPADIC を使用することで最も高い値を得られた. 国内カテゴリに含まれる日本の政治や社会に関する記事は, 未知語を既知の新語や複合名詞で分割してしまうことで悪影響があった考えられる. 他方で, 政治や社会に関する新語や複合語は語義曖昧性の解消に寄与するので適合率が高まったと考えられる. ベストな結果に着目しても適合率と再現率が両方とも他のカテゴリと較べて低いことから, 国内カテゴリの記事をより正しく分類するには単語以外の情報も使って特徴ベクトルを作成する必要がある.

国際カテゴリでは NEologd の適合率が IPADIC を下回っている. このことから国際カテゴリでは名詞以外の単語が語彙の曖昧性解消に重要だと考えられる.

経済カテゴリにおける適合率は, 全ての品詞で特徴ベクトルを作成する場合は UniDic を用 いた場合に最も高かった. このことから経済カテゴリでは新語や複合名詞に埋め込まれやすい一般名詞の一部が分類性能の向上に寄与していると考えられる.

エンタメカテゴリにおいては, NEologd 20161103 で特徴ベ クトルを作成した場合の F 値が最も高い. 他方, それ以前の mecab-ipadic-NEologd では IPADIC より良い結果が出ていない. この差について思い当たる一番大きな原因は人名の扱いである. NEologd 20161103 以前の mecab-ipadic-NEologd ではフルネームが多数登録されている一方で, 名字や名前は網羅的に登録されていなかった. しかし, NEologd 20161103 以降は 3.3.1 節で述べた方法で名字と名前を大量に採録した. さらに登録した人名に関する見出し語についてその悪影響が少なくなる様に, コストの算出方法を 4.5 節で述べた方法で改善した. このことから, エンタメカテゴリでは人名や新語や複合名詞を構成する文字列が, 既存の見出し語によって誤って分割されてしまう場合に対処する必要があると考えられる.

スポーツカテゴリは結果を割愛しているが, 名詞のみで特徴ベクトルを作成すると, 全ての品詞で特徴ベクトルを作成する場合よりも適合率と F 値が向上する. このカテゴリは 8 つのカテゴリの中で最も F 値が高く, mecab-ipadic-NEologd を使った場合は 0.95 を超えている. そのため, 現状では他のカテゴリによって見つかった課題を解決することを優先して問題が無いと考えられる.

IT・科学カテゴリは適合率が UniDic を用いた時に最も高いため, IT・科学カテゴリの未知語に対して, 既知の新語や複合名詞が悪影響を与えやすいと考えられる. 他方で, 再現率の向上には新語や複合名詞の追加が有効であること

も分かる.

ライフカテゴリの分類性能は, 8 カテゴリの中で最もかつ大幅に悪く, 適合率は UniDic を用いた方が高い. ライフカテゴリは新語や未知の複合名詞の影響が強く, 活発に新語や複合名詞を登録するとともに, 未知の形態素を追加することで現状よりも効率よく対処できる分野だと考えられる.

地域カテゴリは明確な分析が難しい. 新語や複合語の追加は再現率の向上に寄与するが, 未知の新語や複合名詞が誤分割されることによる適合率に対する悪影響が出やすいと考えられる. また, 日本全体で共有している文化的な背景から実体の呼び名が決まるため, 全国の異なる地域に全く同じ名前の全く違う実体が存在してしまうことも, 影響していると考えられる.

以上の考察は使用するニュース記事の公開時期を時期を問わず有向であると考えられる. その理由は, 今回の実験の前に異なる時期に同程度の分量の学習データと評価データで大カテゴリ記事セットを作り実験した場合にも同じ傾向の実験結果を得られたからである.

また表 5.2 に実験結果のマクロ平均とマイクロ平均をまとめた.

表 13 大カテゴリ記事セット (30 日分で学習) における実験結果のマクロ平均とマイクロ平均

	マクロ平均			マイクロ平均		
辞書名	適合率	再現率	F 値	適合率	再現率	F 値
IPADIC v2.7.0	0.846	0.808	0.823	0.887	0.887	0.887
UniDic v2.1.2	0.836	0.713	0.745	0.838	0.838	0.838
NEologd 20160919	**0.849**	0.813	**0.827**	0.888	0.888	0.888
NEologd 20161021	0.847	0.813	0.827	**0.888**	**0.888**	**0.888**
NEologd 20161103	0.846	**0.813**	0.827	0.888	0.888	0.888

マクロ平均の違いに着目すると, F 値は NEologd 20160919 を使用した場合の結果が最も良い. この結果は他のすべての辞書との符号検定の結果の P 値≦ 0.001 となり有意水準 0.001 で有意であった. 全体としては NEologd のように新語や複合語を追加することによる不利益は大カテゴリ記事セットの文書分類タスクでは発生しない.

マクロ平均とマイクロ平均 の F 値の違いに着目すると, 上記までの個別の考察で述べた通り, それぞれの分かち書きの辞書には分類性能の向上に十分貢献できていないカテゴリがあると, 全体的にマイクロ平均の方が F 値が高いことからも推察できる.

5.3 学習データ量を変えた場合の分類性能の比較実験

前節では辞書に見出し語を追加していくことの効果を文書分類を例に考えた. 本節では同じく文書分類をする際に学習データを増やしていくことで分類性能を向上させる手法と, 辞書に見出し語を追加していくことの関係について考えるための実験をおこなう.

評価データと使用した学習器やその設定は 5.2 節と同じだが, 学習データとして学習データセットの 1 ヶ月分から 5 ヶ月分までのそれぞれを使用した. 評価データと学習データを分かち書きする際に使用する mecab-ipadic-NEologd を 2016 年 11 月 3 日更新分飲みに絞ったが, 表 5.2 に示した通り, 見出し語を追加することによる不利益が無いため, mecab-ipadic-NEologd についてはより多くの見出し語が含まれているものを採用する.

はじめに表 14 に大カテゴリ記事セットにおける文書分類結果の F 値のマクロ平均とマイクロ平均をまとめる.

いずれのセットを用いた場合においても, F 値のマイクロ平均, マクロ平均ともに NEologd 20161103 を使用した場合の結果が最も良い. この結果は他のすべての辞書との符号検定の結果の P 値≦ 0.001 となっており有意水準 0.001 で有意であった. NEologd 20161103 を使用した場合は 5 ヶ月分のセットから特徴ベクトルを作っても, 1 ヶ月分のセットを使ったときと比べてマイクロ平均, マクロ平均ともに低下していない. 他方, IPADIC や UniDic を使用した場合はマイクロ平均, マクロ平均ともに 1 ヶ月分のセットを使ったときの半分以下まで低下している.

IPADIC や UniDic を使用した際の文書分類性能が高まる可能性を模索することもできたが, 同じ工夫は mecab-ipadic-NEologd でも可能であるため, 今回は素朴な実験結果のままで議論を進める.

IPADIC を使用した場合は学習データの量を増やすことによって, 概ね緩やかに F 値が下がっていった. これは学習自体は上手くできているが, 誤分割による影響が学習事例の増加とともに顕著になったと考えられる. UniDic を使用した場合は学習データの量を増やした際に F 値が激しく上下した. UniDic を使って分かち書きをすると文書は極めて短い形態素に分かち書きされるため, 1 記事を表現するための次元の数が IPADIC や mecab-ipadic-NEologd を使用した場合より高く多くの情報を表現できる. そのため, 学習が上手くいくならデータ量を活かした性能向上の可能性があり, 実際に 4 ヶ月分を使用した時は上手く学習できたのだろう. しかしデータ量を変える度に, 明示的にでも暗黙的にでも最適なパラメタを探索する処理が必要ならそのコストは高いといえる. また, 誤分割による影響は IPADIC と同様に見られる. それに対して, mecab-ipadic-NEologd を使用した場合は, データ量が増えるにつれて一旦性能が上昇し再び低下したが, 上下幅が少なく性能が安定しており, さらにデータ量を増やすことによってより頑健かつ高い分類性能をもったモデルを作成できると考えられる.

次に表 15 に小カテゴリ記事セットについても F 値のマクロ平均とマイクロ平均をまとめる.

小カテゴリ記事セットも大カテゴリ記事セットのときと同様に, F 値のマイクロ平均, マクロ平均ともに NEologd 20161103 を使用した場合に最も良く, 符号検定の結果の P 値≦ 0.001 となっており有意水準 0.001 で有意であった.

表 15 から, IPADIC を雑に使用した場合に驚くほど結果が出ないときには, この表の様な状態になっている可能

表 14 大カテゴリ記事セットにおける文書分類結果の F 値のマクロ平均とマイクロ平均

	学習データに使ったニュース記事の公開日の範囲									
	1 ヶ月分		2 ヶ月分		3 ヶ月分		4 ヶ月分		5 ヶ月分	
辞書名	マクロ平均	マイクロ平均	マクロ平均	マイクロ平均	マクロ平均	マイクロ平均	マクロ平均	マイクロ平均	マクロ平均	マイクロ平均
IPADIC v2.7.0	0.823	0.887	0.827	0.891	0.782	0.863	0.537	0.695	0.208	0.383
UniDic v2.1.2	0.745	0.838	0.411	0.463	0.537	0.626	0.589	0.649	0.363	0.410
NEologd 20161103	**0.827**	**0.888**	**0.839**	**0.897**	**0.844**	**0.900**	**0.839**	**0.900**	**0.833**	**0.898**

表 15 小カテゴリ記事セットにおける文書分類結果の F 値のマクロ平均とマイクロ平均

	学習データに使ったニュース記事の公開日の範囲									
	1 ヶ月分		2 ヶ月分		3 ヶ月分		4 ヶ月分		5 ヶ月分	
辞書名	マクロ平均	マイクロ平均	マクロ平均	マイクロ平均	マクロ平均	マイクロ平均	マクロ平均	マイクロ平均	マクロ平均	マイクロ平均
IPADIC v2.7.0	0.630	0.799	0.463	0.686	0.302	0.539	0.038	0.219	0.002	0.003
UniDic v2.1.2	0.308	0.596	0.105	0.191	0.197	0.310	0.216	0.312	0.118	0.151
NEologd 20161103	**0.656**	**0.809**	**0.674**	**0.822**	**0.675**	**0.818**	**0.639**	**0.793**	**0.562**	**0.743**

性があるということが分かる. データ量が 1 ヶ月分の時は mecab-ipadic-NEologd との差はそれほど大きくないが, 学習データ量を増やした際の性能の低下速度が速い. なぜここまで早く性能が劣化するのかについては, 今後調査する必要がある. UniDic は 1 ヶ月分の学習データを使用している時点から, 誤分割の影響をうけており結果が芳しくない. ただ, 学習データを増やすことによって性能が上下する際の傾向が大カテゴリ記事セットを使用した場合と同様であった. mecab-ipadic-NEologd を使用した場合は学習データの量を 3 ヶ月分まで増やしても, 分類性能は劣化しなかった. しかし, 4 ヶ月以上になるとデータ量に比例して性能が下がっていく. マクロ平均がより早く劣化していくことから, mecab-ipadic-NEologd が対応できていない分野が存在し, 2016 年 6 月から 7 月にかけてその分野に関する記事が多数出現したのではないかと考えられる.

2 つの実験から新語や未知の複合名詞を見出し語として分かち書き用の辞書に追加し続けることで, 辞書に見出し語を追加しない場合や, 短い形態素に区切る場合と比べて, 少なくとも文書分類器の学習データ量を増やした際の性能劣化の速度を緩やかにできることが分かった. 分類器を実サービスに適用する際に, 学習データを足した途端に性能の予測がつかなかったり, 急激に性能が悪化したりすることは非常に困るため, mecab-ipadic-NEologd によって得られたような結果が他の実世界タスクでも実現できるなら, 性能を制御しやすい文書分類器が作れると考えられる.

6. 関連研究

村脇ら [10][11] は未知語問題を解決するために語彙獲得器がテキスト中の未知語を同定して, 人手の介在なしに解析用辞書に追加する手法を提案しており非常に参考になる. 我々も新語と未知語の監視をするプロセスが検出した低頻度なキーワードに関しては, 採録の判断をできる限り自動化したいと考えているが, 村脇らほどの瞬間的な語彙の獲得は目指していない. また, 分かち書き用辞書を実用するうえで採録すべき語彙を極力自動で採録したいと考えているが, 即時性よりは網羅性と確度を重視したいと考えている.

7. おわりに

我々は mecab-ipadic-NEologd という分かち書き用辞書を用いて形態素より長い固有名詞や複合語を一語として分かち書きすること, 及び, 辞書生成システム NEologd を運用して短い周期で更新することの各効果を, ニュース記事を複数のカテゴリに分類する実験を通じて確認し, mecab-ipadic-NEologd が有効であることを示した. 具体的には, 語彙を辞書に継続して追加することによる不利益がとくに無く, また学習データを増やすことが原因である分類性能の劣化速度を緩やかにし, 分類器を実用する際に性能を制御しやすい文書分類器を作れることが分かった.

今回は実験に使用していないが,NEologd の 4 つ組リストでは, 人名や組織名, 地名などは区別がついているので, 固有名詞の品詞情報を付与していることの効果についても調査したい.

参考文献

[1] 森信介, ニュービッググラム. 言語資源の追加:辞書かコーパスか, 情報処理学会研究報告, 自然言語処理研究会報告 2014-NL-216(12),pp.1-3,2014-05-15
[2] 浅原正幸, 松本裕治.ipadic version 2.7.0 ユーザーズマニュアル,2003
[3] IREX 実行委員会 (編). IREX ワークショップ予稿集. IREX 実行委員会,1999.
[4] Satoshi Sekine. Extended named entity ontology with attribute information. In In Proceedings of the 5th International Conference on Language Resources and Evaluation, 2008.
[5] Satoshi Sekine and Chikashi Nobata. Definition, dictionary and tagger for extended named entities. In Proceedings of the Forth International Conference on Language Resources and Evaluation, 2004.
[6] Satoshi Sekine, Kiyoshi Sudo, and Chikashi Nobata. Extended Named Entity Hierarchy. In LREC2002, 2002.
[7] 工藤拓. Mecab: Yet another part-of-speech and morphological analyzer. http://taku910.github.io/mecab/
[8] Graham Neubig, 中田 陽介, 森 信介. 点推定と能動学習を用いた自動単語分割器の分野適応. 言語処理学会第 16 回年次大会 (NLP2010), 2010.
[9] 森田一, 黒橋 禎夫. RNN 言語モデルを用いた日本語形態素解析の実用化. 情報処理学会 第 78 回全国大会, 2016
[10] Yugo Murawaki and Sadao Kurohashi. Online acquisition of Japanese unknown morphemes using morphological constraints. In Proc. of EMNLP 2008, pp. 429-437, 2008.
[11] 村脇有吾, 黒橋禎夫. 語彙獲得のための過分割未知語の検出. 言語処理学会第 15 回年次大会 発表論文集, pp.324-327, 2009.