単語分かち書き辞書 mecab-ipadic-NEologd の実装と情報検索における効果的な使用方法の検討

佐藤 敏紀† 橋本 泰一† 奥村 学‡
LINE 株式会社 Data Labs†
東京工業大学 科学技術創成研究院 未来産業技術研究所‡
{overlast, taiichi.hashimoto}@linecorp.com†, oku@pi.titech.ac.jp‡

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のおもな課題は, 単語間の境界を推定する処理の精度や, 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. 実サービスのテキストの分かち書き結果を応用する場合, 必ず解決すべき問題は, 未知の単語・形態素が原因の解析誤りによる性能の低下である. この問題はチャンカーの作成や, 後処理による目立つ解析誤りパターンの修正, 個別の解析誤り事例への対処などで解決できたが, 案件やデータが変わる度にその様な作業を繰り返す必要があった.

その様な背景から我々は, 単語分かち書き用の辞書生成に必要な資源を収集するための NEologd というシステムを運用し, 単語分かち書き用の辞書である mecab-ipadic-NEologd を継続的に更新している. これにより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を大幅に軽減できると考えられる.

本稿では単語分かち書き用の辞書である mecab-ipadic-NEologd を実装し, 継続的に更新することによって得た知見と, この辞書を情報検索向けに応用する際に必要な技術や実装に関する考察について述べる.

2 mecab-ipadic-NEologd の実装

2.1 システムによる Web 上からの語彙収集

NEologd[1] というシステムを使った語彙獲得タスク [1] では表 1 に示す「表層」,「読み仮名」,「表層の原型」,「品詞情報」の 4 つの要素の集合 (以降, 4 つ組と呼ぶ) を収集している. 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で構成されており, 収集したデータから自動または半自動的に表 2 に示す 4 つ組のリスト (以降, 4 つ組リストと呼ぶ) を生成する.

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

4 つ組を作るための語彙獲得に関わる処理は大きく「新語や未知語の検出」,「Web サイトのクロール」,「語彙が不足しているドメインに属する用語の網羅」,「テンプレートによる生成」,「ホワイトリスト, ブラックリストの管理」の 5 種類に分けられ, 必要性と効果が共に高い時は人手での作業も行う. NEologd で 4 つ組を収集する際の判断基準や, 語彙の獲得に関わる処理の詳細は [1] を参照されたい. 我々は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.

2.2 単語分かち書き用辞書の生成

mecab-ipadic-NEologd[2] は文を形態素単位に分かち書きするための辞書ではなく, 固有名詞や複合名詞などの長い単語を 1 単語として分かち書きするための MeCab[3] 用の辞書である. この辞書は NEologd の 4 つ組リストとその他の人手で蓄積した 4 つ組リストを用いて生成される. 内部的には MeCab 用の IPA 辞書[3](以下 IPADIC と呼ぶ) を基本語彙辞書として扱っており, 基本語彙辞書のみで正しく単語分かち書きできる場合になるべく悪影響を与えない様に実装している. また, 解析対象の文として Web 上のニュース記事や SNS 上の発言などを想定している.

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

[2] https://github.com/neologd/mecab-ipadic-neologd

[3] http://taku910.github.io/mecab/

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,*,*
平成 31 年	ヘイセイサンジュウイチネン	2019 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,*,*,*

mecab-ipadic-NEologd を実用する過程で「コーパスを使わない単語生成コストの調整」,「採録が必要な言葉のタイプや必要な更新頻度の特定」,「基礎語彙辞書が原因であるエラーの解消」,「自動獲得した新語から人名を検出」の 4 つの課題は, 継続的に結果を改善することで辞書全体の性能が着実に向上したので, 以降の節でこれらの課題の詳細を述べる.

2.2.1 コーパスを使わない単語生成コストの調整

現状では mecab-ipadic-NEologd を生成する際に, 分かち書きタスクにおける有効性の高さや開発速度, メンテナンスの難易度を考慮してタグ付きコーパス無しで単語生成コストを調整している. 辞書に採録する見出し語数が増える過程で, ある見出し語の一部と共通する部分文字列を持つ別の見出し語があるときに, 前者が原因で後者が分割される問題を解決するために採用したコストの調整法を手続き 1 に示す.

この手続きのインターフェイス関数である MAKE-NEOLOGD-DICT-SEED は, NEologd 形式の 4 つ組リストを引数に取り, 第 77 行目で任意に処理して公開可能な見出し語辞書データを出力する.

4 つ組の表層の単語生起コストを段階的に調整しているのは関数 GET-TUNED-CSV-FILE である. コストの初期値は第 4〜11 行内で 4 つ組の品詞情報ごとに異なる任意の手法により決める. 具体的には, 基礎語彙辞書の品詞・文字列長ごとの形態素生起コストの分布を加味する処理や, 名詞系エントリの場合は基礎語彙辞書を用いて得た「見出し語が分割されない生起コスト」から表層の文字列長に比例した値を差し引く処理などを実行している.

コストの正しさは関数 GET-BOUNDARY-NUM によって判定する. テスト用の事例は「運用時に分かち書き誤りを検出した文脈」を基に作ったテンプレートに見出し語を埋め込み生成する. 埋め込んだ見出し語が複数の単語に分かち書きされた場合, 特殊な事例を除いて最大の分かち書き数が得られる. 第 54〜61 行で算出した見出し語の分割数の正解率が $min\_accuracy$ を達成できなければ, 関数 FIX-OCCUR-COST が呼ばれ, 見出し語が $mecab\_index$ によって分割されないコストを任意の手法で算出する. この処理は $min\_accuracy$ 以上の性能を達成するまで複数回実行されるので, 各回の調整は小幅で良い. $min\_accuracy$ はテスト用の事例の量や難易度を鑑みて調整する.

第 66〜70 行では調整が不要であった CSV データと調整した CSV データを結合しており, 再び第 44 行に達した際に更新した $mecab\_csv\_file$ で MeCab のシステム辞書が作られる. 現在の運用では, 以上の while 文の処理を 3〜4 回程度繰り返すことで調整が完了する.

表 3: 頻繁な採録が必要であった言葉のタイプ

タイプ名	生成方法	解説や詳細なタイプの例
人名	自動	芸能人, 著名人, 専門家
作品名	自動	書籍, 曲, 映画, アニメ, ゲーム
キャラクター名	自動	作品中の想像上の人物・キャラ
IT・コンピューター	自動	製品, サービス, CS の専門用語
検索キーワード	半自動	複数サイトで検索頻度が上位な語
新語・流行語	半自動	ニュース・SNS で高頻度な語
ハッシュタグ	半自動	SNS で高頻度なハッシュタグ

2.2.2 採録すべき言葉のタイプと更新頻度

mecab-ipadic-NEologd の更新を継続する過程で, 語彙の不足や新語・未知語の出現に効率よく対処するために優先的な採録が必要になる言葉のタイプを知る必要があった. 未採録かつ採録すべき言葉 (名詞系以外も含む) を検知した際に, その言葉を含むタイプが未知の場合は, そのタイプの範囲内の言葉の集合を NEologd で収集, もしくは人手で作り, その集合から 4 つ組リストを生成してきた. 以下の表 3 から表 5 に現状までに優先的に採録してきた言葉のタイプをまとめた.

表 3 から表 5 の範囲にあるタイプの言葉に関する 4 つ組リストを対応する採録頻度で更新すると共に, 表 6 にタイプに含まれる言葉は, 新語や未知語として検出される前に網羅する様に務める必要があった.

「英数字のみの略語」や「業界用語」の範囲に含まれる言葉は, タイプが異なると読み方や意味が変わる

手続き 1 MAKE-NEOLOGD-DICT-SEED

```
Input:
    neologd_list は NEologd 形式の 4 つ組リスト
Output:
    neologd_seed は MeCab 用の辞書形式な見出し語辞書の CSV ファイル
 1:
 2: function GET-MECAB-CSV-FILE(neologd_list)
 3:     mecab_csv_file は新しいファイルとする
 4:     for i ← 1 to neologd_list.length do
 5:         entry = neologd_list[i] // i 番目の 4 つ組
 6:         surface = entry 中の表層の文字列
 7:         pos = mecab_csv 中の品詞情報
 8:         cost = pos ごとに異なる任意の手法で surface の単語生起コストを得る
 9:         csv = entry と cost から MeCab 形式の CSV データを生成
10:         mecab_csv_file に csv を追記する
11:     end for
12:     return mecab_csv_file
13: end function
14:
15: function GET-BOUNDARY-NUM(mecab_index, mecab_csv)
16:     surface = mecab_csv 中の見出し語の文字列
17:     pos = mecab_csv 中の品詞情報
18:     test_case_list = surface と pos から複数のテスト事例を生成
19:     mecab = mecab_index をシステム辞書に使って起動した MeCab
20:     bound_num = 0
21:     for i ← 1 to test_case_list.length do
22:         mecab_result = mecab で test_case_list[i] を処理
23:         tmp_num = mecab_result から surface の分割数を獲得
24:         if bound_num < tmp_num then bound_num = tmp_num
25:         end if
26:     end for
27:     return bound_num
28: end function
29:
30: function FIX-OCCUR-COST(mecab_index, csv_list)
31:     fixed_csv_list は新しい配列とする
32:     mecab = mecab_index をシステム辞書に使って起動した MeCab
33:     for i ← 1 to csv_list.length do
34:         target_csv = csv_list[i]
35:         new_cost = mecab で target_csv の見出し語の単語生起コストを再計算
36:         fixed_csv = target_csv の単語生起コストを new_cost で置換
37:         push fixed_csv to fixed_csv_list
38:     end for
39:     return fixed_csv_list
40: end function
41:
42: function GET-TUNED-CSV-FILE(neologd_list)
43:     mecab_csv_file = GET-MECAB-CSV-FILE(neologd_list)
44:     mecab_index = mecab_csv_file を基に MeCab のシステム辞書を作る
45:     true_csv_list と false_csv_list は 2 つの新しい配列とする
46:     accuracy = -1.0
47:     min_accuracy = 目標とする最低正解率 (0.0〜1.0)
48:     while accuracy < min_accuracy do
49:         true_case_num = 0
50:         false_case_num = 0
51:         for i ← 1 to neologd_list.length do
52:             mecab_csv = mecab_csv_file の i 行目
53:             b = GET-BOUNDARY-NUM(mecab_index, mecab_csv)
54:             if b == 0 then
55:                 push mecab_csv to true_csv_list
56:                 true_case_num++
57:             else
58:                 push mecab_csv to false_csv_list
59:                 false_case_num++
60:             end if
61:         end for
62:         accuracy= true_case_num / neologd_list.length;
63:         if accuracy >= min_accuracy then
64:             return mecab_csv_file
65:         end if
66:         csv_file は新しいファイルとする
67:         csv_file に true_csv_list を追記
68:         fixed_list = FIX-OCCUR-COST(mecab_index, false_csv_list)
69:         csv_file に fixed_list を追記
70:         swap mecab_csv_file for csv_file // 更新した CSV ファイルと入れ替え
71:     end while
72:     return mecab_csv_file
73: end function
74:
75: function MAKE-NEOLOGD-DICT-SEED(neologd_list)
76:     tuned_mecab_csv_file = GET-TUNED-CSV-FILE(neologd_list)
77:     neologd_seed =tuned_mecab_csv_file から公開可能な見出し語辞書を作成する
78:     return neologd_seed
79: end function
```

場合がある (2016 年末に流行した曲名・略語である「PPAP(ピーピーエーピー)」ですら既存の略語が存在している) ため, 見出し語として採録する場合に地味で特別な対応が必要になる場合があった. 「方言」や「隠語」が混じった日本語テキストを扱う技術は強く必要とされているが, 扱う日本語の難易度が我々の想定よりも高くなることから一旦採用を見送っている.

表 4: 定期的な採録が必要であった言葉のタイプ

タイプ名	生成方法	解説や詳細なタイプの例
住所	自動	テンプレートで異表記を生成
駅・ランドマーク	自動	駅, 名所, 待ち合わせ場所
組織名	自動	株式会社, 学校, 病院, チーム
ファッション	自動	ブランド名, 衣料品, 服飾雑貨
有名チェーン店	自動	飲食店名, 量販店名
人名の名前	自動	難読または稀有な名前以外
食事・料理	半自動	料理名, 食品名, 食材名
医療・ヘルスケア	半自動	病名, 薬品名, 身体の部位
時間・数値表現	半自動	パターン生成する際に範囲指定
複合名詞	半自動	検出した未知語から人手抽出
サ変名詞	半自動	検出した未知語から人手抽出
形容詞	半自動	他の辞書や SNS を観察し生成
形容動詞	半自動	他の辞書や SNS を観察し生成
副詞	半自動	他の辞書や SNS を観察し生成
感動詞	半自動	パターン生成する際に範囲指定
Unicode 絵文字	半自動	Unicode の正式リリース後に
顔文字	半自動	監視先での出現頻度が高いもの
若者言葉・オタク用語	人手	悪影響がない語のみ採録

表 5: 採録がほぼ不要であった言葉のタイプ

タイプ名	生成方法	解説や詳細なタイプの例
人名の姓	自動	難読または稀有な名前以外
決まり文句	自動	四字熟語, ことわざ, 慣用句
動物・植物名	半自動	新種や未知の生物が出てきたら
神社・寺院名	半自動	新しく創建・建立されたら
島・河川・山脈名	半自動	未知の流域や領域があれば

2.2.3 基礎語彙辞書が原因であるエラーの解消

基礎語彙辞書である IPADIC の名詞系エントリに 2.2.1 節で述べたコストの調整を行った. その結果「日本酒」や「義務教育」など 10168 エントリのコストが調整された. また, 「人民元」や「不可逆」の様に付与された読み仮名が誤っているエントリの候補を検出し人手で精査した後, 誤りを訂正した. 読み仮名の誤りはまだ多く含まれているが今後も継続的に訂正する.

2.2.4 自動獲得した新語からの人名の検出

NEologd が収集した固有名詞の 4 つ組リストのうちタイプが未整理なものを最新の mecab-ipadic-NEologd で解析して, 「姓, 名」または, その逆に分かち書きされた表層を自動的に人名扱いすることで多少誤りはあるが, 効率良く品詞情報を付与できた.

表 6: 網羅する作業を継続すべき言葉のタイプ

タイプ名	生成方法	解説や詳細なタイプの例
歴史上の著名な人名	自動	信頼できる資源から網羅的に生成
学問	自動	範囲を決めて網羅的に収集
乗り物	自動	電車, 自動車, バイク, 船, 飛行機
文化	自動	芸術, 芸能, 文学に関する用語
日常生活	半自動	一般市民が生活の中で使う言葉
社会や経済, 経営	半自動	節目や緊急の出来事に関する語
医学	半自動	より専門的な医療用語
珍しい動物・植物名	半自動	外来種や既知の生物の別名
スポーツ	半自動	範囲を決めて網羅的に収集
趣味	半自動	範囲を決めて網羅的に収集
その他	人手	未採録だと大きな悪影響がある語

表 7: MeCab & mecab-ipadic-NEologd による N-Best 解の 1 位に N-Best 解中の任意の単語を埋め込んだ結果

表層	品詞情報	原型	読み仮名
任天堂	名詞, 固有名詞, 組織,*,*,*	任天堂	ニンテンドウ
は	助詞, 係助詞,*,*,*,*	は	ハ
『	記号, 括弧開,*,*,*,*	『	『
ゼルダの伝説ブレスオブザワイルド	名詞, 固有名詞, 一般,*,*,*	ゼルダの伝説 ブレス オブ ザ ワイルド	ゼルダノデンセツブレスオブザワイルド
ゼルダの伝説	**名詞, 固有名詞, 一般,*,*,***	**ゼルダの伝説**	ゼルダノデンセツ
ゼルダ	**名詞, 固有名詞, 人名, 一般,*,***	**ゼルダ**	**ゼルダ**
ブレス	**名詞, 固有名詞, 一般,*,*,***	**ブレス**	**ブレス**
オブ	**名詞, 固有名詞, 一般,*,*,***	**オブ**	**オブ**
ザワイルド	**名詞, 固有名詞, 一般,*,*,***	**ザ・ワイルド**	**ザワイルド**
ワイルド	**名詞, 形容動詞語幹,*,*,*,***	**ワイルド**	**ワイルド**
』	記号, 括弧閉,*,*,*,*	』	』
を	助詞, 格助詞, 一般,*,*,*	を	ヲ
3 月 3 日	名詞, 固有名詞, 一般,*,*,*	3 月 3 日	サンガツミッカ
3 月	**名詞, 固有名詞, 一般,*,*,***	**3 月**	**サンガツ**
3 日	**名詞, 固有名詞, 一般,*,*,***	**3 日**	**ミッカ**
に	助詞, 格助詞, 一般,*,*,*	に	ニ
Nintendo Switch	名詞, 固有名詞, 一般,*,*,*	Nintendo Switch	ニンテンドースイッチ
Nintendo	**名詞, 固有名詞, 組織,*,*,***	**任天堂**	**ニンテンドウ**
Switch	**名詞, 固有名詞, 一般,*,*,***	**Switch**	**スイッチ**
と	助詞, 並立助詞,*,*,*,*	と	ト
同時	名詞, 一般,*,*,*,*	同時	ドウジ
発売	名詞, サ変接続,*,*,*,*	発売	ハツバイ
し	動詞, 自立,*,*, サ変・スル, 連用形	する	シ
た	助動詞,*,*,*, 特殊・タ, 基本形	た	タ
。	記号, 句点,*,*,*,*	。	。

3 情報検索における有効性の考察

mecab-ipadic-NEologd は語彙数が 5,533,854 語 (2016 年 11 月 3 日時点) と他の辞書より多く, 互いに重複した部分文字列をもつエントリも多く含む. 我々は文書分類タスクにおけるこの辞書の有効性の高さを, ニュース記事のカテゴリ分類実験を通じて確認した [1]. 他方, 全文検索における転置索引の作成やクエリ処理において mecab-ipadic-NEologd のみを素朴に使った場合, 索引作成に使う文書と文書を検索するクエリ文字列との双方を分かち書きした結果が揺れるため, 分かち書き結果 (N-best 解の 1 位) の一貫性が低くなり [4], UniDic [4] の様な短い形態素単位を単語として扱う辞書と比べて検索の再現率が低くなる.

ここで, 検索漏れに関する問題のみに対処することを考えると, 達成すべき目標は「解析過程で利用しているラティスから状況に合わせて都合よく文字列を抽出し, N-Best 解の 1 位の分かち書き結果に埋め込むこと」だとも考えられる. 上記の「都合」は, 理想を考えるとデータやログから決めるべきだが, 一般にはより生起コストが低く文字列長が長い単語の境界を跨がない分かち書き結果を再帰的に得れば十分である.

例文「任天堂は『ゼルダの伝説ブレスオブザワイルド』を 3 月 3 日に Nintendo Switch と同時発売した. 」を, mecab-ipadic-NEologd で解析する過程のラティス構造から, より長い単語と読み仮名が部分一致した固有名詞を抽出し, より長い単語の直後への挿入処理を擬似的に行った場合を表 7 に示す. この例では, IPADIC の見出し語以外に, NEologd で収集・生成した固有名詞や日付表現を取得できている. mecab-ipadic-NEologd を用いて分かち書きした場合, 採録されている語に関しては改めて正しい単語境界を推定する必要がなく, 表 7 の様な結果は素朴な手法でも実際に獲得できる.

4 おわりに

本稿では, mecab-ipadic-NEologd を継続的に更新し続けたことで得た知見として, コーパス無しで単語生起コストを調整する方法と, 採録する必要があった言葉のカテゴリと必要な採録頻度について述べた. また, この辞書を検索向けに応用する際に必要な実装に関する考察をした. 我々は今後, mecab-ipadic-NEologd を検索向けに素朴に使うためのツールを実装し, 考察した課題に対処する予定である.

参考文献

[1] 佐藤敏紀, 橋本泰一, 奥村学. 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 —, 情報処理学会 自然言語処理研究会研究報告 2016-NL-229-15

[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] 工藤拓. Mecab: Yet another part-of-speech and morphological analyzer.

[4] 高橋文彦, 颯々野学. 情報検索のための単語分割一貫性の定量的評価. 第 22 回言語処理学会年次大会, 2016.

[4] http://pj.ninjal.ac.jp/corpus_center/unidic