Vol.2020-MPS-130 No.6
2020/9/29

単語の分散表現に基づく流行語分析

堺 雄之介[1,a)]　伊東 栄典[2,b)]

概要：Twitter 等の SNS や，利用者が動画や小説等を投稿する CGM では消費者の生の声が反映される．そのため，消費者の嗜好や流行の分析対象に適している．流行語の抽出手法として単純に文書中の単語出現頻度を用いる手法や，流行語抽出対象とする期間と，その前の期間での出現頻度の差分を取る手法がある．本研究では流行語抽出の新たなアプローチとして，単語の類似語に注目した手法を提案する．また提案手法を可視化する流行語分析ツールを作成した．実際に「syosetu.com」の小説メタデータ群に，提案手法とツールを適用した．本論文では，提案手法，ツール，および適用結果について述べる．

キーワード：CGM, Popular word, trend, fastText, Django

Yunosuke Sakai[1,a)]　Eisuke Ito[2,b)]

Abstract: Social networking services such as Twitter, and CGMs (Consumer Generated Medias) are reflecting the real voices of consumers. Therefore, they are suitable for the analysis of consumer preferences and trends. There are two methods of extracting popular words: one is to simply use TF (term frequency) of occurrence of the words in the document, and the other is to take the difference of TF between a paticulary period and the next period. In this study, we propose a new approach to extracting popular words by focusing on the similarity of words. We also developed a buzzword analysis tool to visualize the proposed method. We actually applied the proposed method and tools to the novel metadata group on "syosetu.com". In this paper, we describe the proposed method, tools, and results of the application.

Keywords: CGM, Popular word, trend, fastText, Django

1. はじめに

その年に世間を賑わせた単語やフレーズ，コンテンツ等が毎年発表されている [1]．2019 年の流行語として「ONE TEAM」,「軽減税率」などが挙げられる．タピオカ入りのドリンクを飲む行為を指す「タピる」に代表されるように，消費者の購買行動を反映した言葉もある．そうした流行語を分析することは商機に繋がる．

近年，動画共有サイトの Youtube やニコニコ動画，Twitter や Facebook 等の SNS のように，一般消費者により制作・提供されるコンテンツを投稿するサービス（CGM, Consumer Generated Media）が人気である．これらのサービスは消費者の声や嗜好が直接共有されるため，大衆の動向を把握するための流行語抽出が行われている．

本研究では CGM の内，小説を投稿するサービス「小説家になろう」での流行語抽出を行った [2].「小説家になろう」では小説のメタデータを対象に分野ごとかつ月ごとの流行語を抽出した．流行語抽出の過程で 2 つの方法を適用し比較した．1 つ目は単語の単純な出現頻度をカウントしたもの，2 つ目は単語の分散表現によって抽出した類似語とその類似度を考慮したものである．

流行語抽出の結果を可視化するためのツールを作成した．このツールは検索エンジンも利用している．「小説家になろう」の小説メタデータに対し，このツールを適用した．

本文の構成を述べる．第 2 章では「小説家になろう」，及び本研究で使用したなろう小説 API について説明する．第 3 章では単語の分散表現取得の手法とデータ処理の流れを述べる．第 4 章では流行語抽出に用いた 2 つの方法の詳細を述べる．第 5 章では「小説家になろう」の小説メタデータ群に，2 つの方法を適用した流行語抽出の結果と考察を述べる．第 6 章では作成した「小説家になろう」の流行語分析ツールについて述べる．第 7 章で関連研究について述べ，第 8 章でまとめと今後の課題を述べる．

[1] 九州大学大学院システム情報科学研究院
[2] 九州大学情報基盤研究開発センター
[a)] y.sakai.a96@s.kyushu-u.ac.jp
[b)] ito.eisuke.523@m.kyushu-u.ac.jp

図 1　小説家になろうトップページ

2.　利用データセット

本研究では流行語分析の対象として，なろう小説 API から取得した「小説家になろう」の小説メタデータを用いた．この章では API とサービスの概要を述べる．

2.1　サービス概要

「小説家になろう」 [3] は，株式会社ヒナプロジェクトが提供する小説投稿サイトである．サイトのトップページ*1を図 1 に示す．利用者登録したユーザーはサイトに小説を投稿でき，公開されている小説は誰でも読むことが出来る．2004 年のサイト開設当初は個人サイトとして運営されていた．その後のアクセス増加により，2008 年からグループによる運営に移行し，2010 年に正式に法人化した．Wikipedia [4] によると，2019 年 4 月時点でアクセス数が月間約 20 億，ユニークユーザーが約 1400 万人である．また 2020 年 1 月 17 日，登録者数は 1,717,219 人，掲載小説数は 703,600 である．

このサイトの小説は「なろう小説」と呼ばれている．「なろう小説」の内，人気が出たものは紙の小説として出版されたり，マンガやアニメの原作になることもある．

「小説家になろう」では小説を閲覧する前に，小説のメタデータが目に入る．そのためユーザーは小説を閲覧するかどうかをメタデータで判断することが少なくない．メタデータにはタイトル，ジャンル，あらすじなどが含まれる，本研究においては小説のあらすじから流行語を抽出する．

2.2　なろう小説 API

なろう小説 API [5] は「小説家になろう」に掲載されている小説のメタデータを取得出来る Web API である．HTTP で GET リクエストを行うと JSON もしくは YAML 形式でデータを取得出来る．この API の出力として小説名やあらすじなど，計 40 項目のデータが得られる．得られるデータ項目の一覧を表 1 に示す．本研究では 2004 年 4 月 20 日から 2019 年 11 月 15 日の期間に投稿された 693,304 件の小説のメタデータを用いた．

*1　https://syosetu.com/

2.3　メタデータ収集

Web API を用いて，全小説のメタデータを取得するクローラーを Python 言語で作成した．クローラーの概要を図 2 に示す．

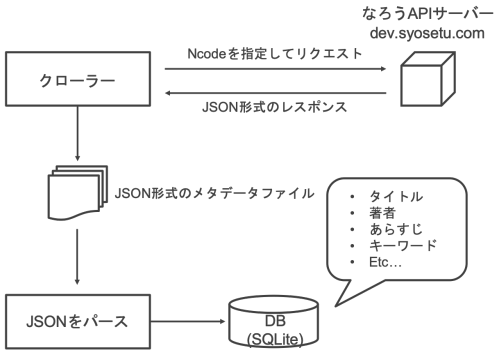

図 2　メタデータクローラー

なろう小説 API へ GET リクエストを行う際，小説毎に割り当てられている識別子 Ncode を指定すれば，特定の小説メタデータを収集できる．Ncode は n で始まり数値 4 桁の後ろにアルファベットが辞書順に付与されている．例えば，n9999ab の次の Ncode は n0000ac となる．今回は n0000a から n2643fw の Ncode を持つ小説メタデータを収集した．

2.4　収集したデータの基礎分析

2.3 節で収集した全データの内，削除されている小説を取り除くと，取得したメタデータ件数は 693,304 である．これは収集を行った 2019 年 11 月 15 日時点での全小説件数である．データ収集で得たデータの概要を表 2 に示す．収集したデータは SQLite [6] を用いてデータベースに格納した．

「小説家になろう」への月ごとの小説投稿数を図 3 に示す．図 3 のグラフは 2.3 節に記載したクローラーが収集したデータを整形し，分析して得たものである．新規小説の投稿数を見ると，短期的な減少はあるものの長期的には単調増加を続けており，現在は約 10,000 作品が毎月新規に投稿されている．

表**??**中の作品は全てファンタジージャンルとして投稿された作品である．投稿される小説にファンタジージャンルのものが多いのも，その人気が反映された結果と考えられる．図 4 に示すように，ノンジャンルを除けばハイファンタジー [ファンタジー] ジャンルに最も多くの作品が投稿されている．

表 1　なろう小説 API から得られる小説メタデータ項目

要素	説明	要素	説明
allcount	全小説出力数	isgl	ガールズラブ：1，それ以外：0
title	小説名	iszankoku	残酷な描写あり：1，それ以外：0
ncode	N コード	istensei	異世界転生：1，それ以外：0
userid	作者のユーザ ID	istenni	異世界転移：1，それ以外：0
writer	作者名	pc_or_k	携帯：1，PC：2，PC と携帯：3
story	小説のあらすじ	global_point	総合評価ポイント
biggenre	大ジャンル	daily_point	日間ポイント
genre	ジャンル	weekly_point	週間ポイント
gensaku	現在未使用項目	monthly_point	月間ポイント
keyword	キーワード	quarter_point	四半期ポイント
general_firstup	初回掲載日	yearly_point	年間ポイント
general_lastup	最終掲載日	fav_novel_cnt	ブックマーク数
novel_type	連載：1, 短編：2	impression_cnt	感想数
end	短編・完結済：0，連載中：1	review_cnt	レビュー数
general_all_no	全掲載部分数	all_point	評価点
length	小説文字数	all_hyoka_cnt	評価者数
time	読了時間	sasie_cnt	挿絵の数
isstop	長期連載停止中：1，それ以外：0	kaiwaritu	会話率
isr15	R15：1，それ以外：0	novelupdated_at	小説の更新日時
isbl	ボーイズラブ：1，それ以外：0	updated_at	最終更新日時

表 2　収集データ概要

項目	内容
期間	2004 年 4 月 20 日〜2019 年 11 月 15 日
形式	json 形式
データ件数 (小説数)	693,304
時点	2019 年 11 月 15 日時点

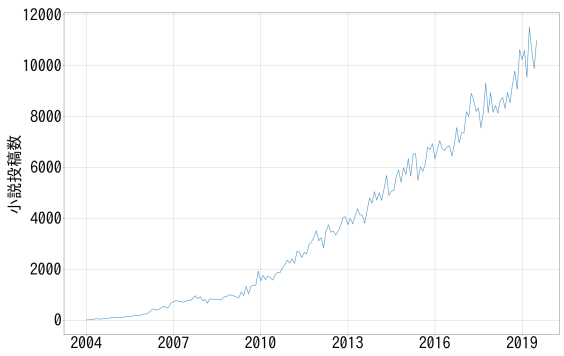

図 3　月ごとの小説投稿数

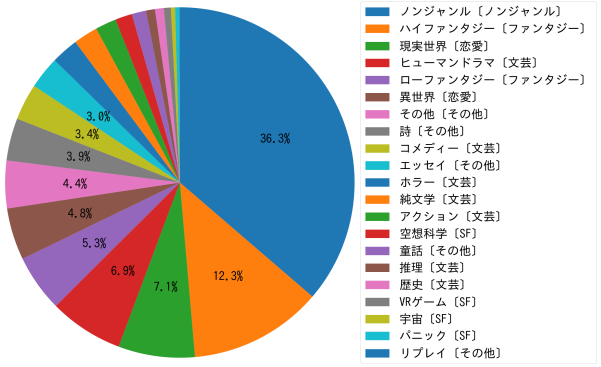

図 4　小説家になろう投稿ジャンルの割合

3. 単語の分散表現

本研究では単語の分散表現を用いて，単語の類似語と類似度を算出する．この章では単語の分散表現と，その取得までのデータ処理の流れを述べる．

3.1 分散表現取得の手法

単語をベクトルとして表現する分散表現を得る手法として Word2Vec [7] がある．Word2Vec は Tomas Mikolov らの開発した分散表現を生成する手法で，各単語を高次元のベクトルで表現する．Word2Vec では，文章中に含まれる単語の出現数を利用する Countinuous Bag-of-Words（CBOW）モデルと，文章中に含まれる単語の並びから単語の出現確率を利用する Skip-gram モデルの両方の学習モデルを用いて，Hierarchical Softmax 及び Negative Sampling によって高速化を行う．

fastText [8] は Facebook AI Research が 2016 年に開発した自然言語処理向けアルゴリズムである．fastText のライブラリでは単語の分散表現のほか，テキスト分類を行うことが可能である．fastText も Word2Vec の開発者の一人である Tomas Mikolov によって開発されているため，Word2Vec 同様に CBOW モデルと，Skip-gram モデルの両

方の学習モデルを利用することができる．また，fastTextという名前の通り Word2Vec を含む他のアルゴリズムに比較して，動作が軽く速いのが特徴である．さらに精度についても向上している．

3.2 データ処理

データ処理の流れを以下と図 5 に示す．本研究ではfastText を用いて単語の分散表現を獲得する．

(1) 各小説のあらすじ（Yahoo!知恵袋の場合は質問文）を形態素解析ツール Mecab [9] で分かち書き文に変換する．
- 新語対応のため形態素解析に IPA-Neologd 辞書 [10] を用いる．
- Mecab での解析の際，分かち書き文に残す品詞を制限する．流行語は名詞が多いため，今回は名詞と固有名詞のみに制限する．

(2) 分かち書き文書群をコーパスとして fastText に入力し，単語の分散表現を得る．
- 分散表現（ベクトル）の次元数は 300 次元とした．

(3) 分かち書き文書群から単語の出現頻度 (TF) を得る．TF のカウントには scikit-learn[11] を用いる．

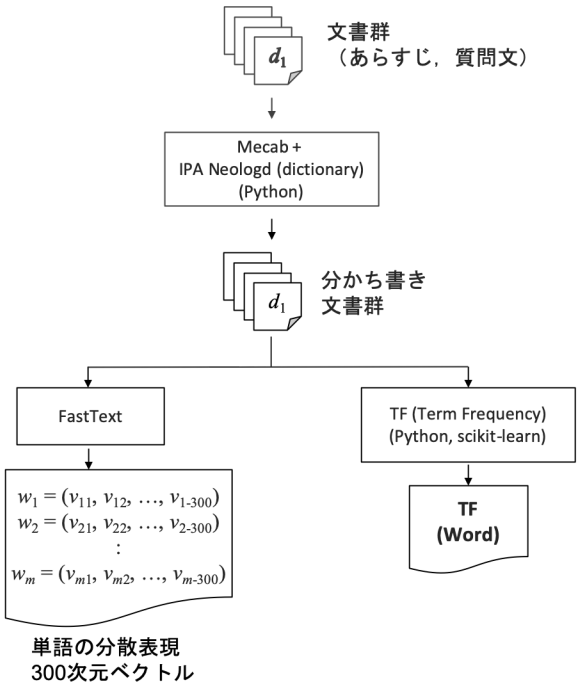

図 5 データ処理の流れ

4. 流行語の抽出手法

ある文書群から単語単位で流行語を抽出しようとする際，最も単純で直感的な方法は各単語の出現頻度を数え上げることである．しかしその方法では流行語を分析する人が求める流行語を抽出出来ない．単純な出現頻度が大きいのは日本語では「の」や「です」などの単語で，英語では「a」や「the」等の単語になる．そのため流行語の抽出対象とする文書から，流行語分析にはふさわしくない単語を取り除く必要がある．このような取り除く単語を不要語，またはStop word という．また数え上げる単語の品詞の限定も解決策の一つである．

抽出対象の文書群にタイムスタンプが付いている場合，出現頻度が急激に大きくなる単語に注目する時系列に基づく分析手法も存在する．特定の単語の一定期間における出現頻度を数え，これを単語の出現速度とする．出現速度の上昇率が著しく大きいものを流行語とする手法である．

本研究では単語の出現頻度を数え上げる方法に加え，後述する類似単語の出現頻度を考慮する方法を検討する．この章ではその 2 つの手法について述べる．どちらの手法も名詞のみを流行語の抽出対象とする．

4.1 方式 1:出現頻度のみ

方式 1 は単語の出現頻度のみで流行語を決めるもので，従来から用いられている素朴な方式である．小説のあらすじ，Yahoo!知恵袋の質問文などの文書集合について，期間 p における単語 w の出現頻度 $tf(w,p)$ を求める．期間 p の出現頻度 $tf(w,p)$ が上位となる単語が，その期間の流行語である．

4.2 方式 2:類似単語の出現頻度を考慮

方式 2 として，類似単語の出現頻度も考慮する方法を提案する．1 つの物事を表す単語が 1 つしかない場合は少ない．省略語や類似する単語などで表現される場合が多い．例えば「オリンピック」と同様の単語に「Olympic」や「五輪」がある．意味的に近い単語に「オリパラ」がある．「オリンピック」の頻度に，類似単語の「Olympic・五輪・オリパラ」の頻度を加えることで，「オリンピック」の流行度をより良く計れるのではないかと考えた．

方式 2 では，期間 p における単語 w の出現頻度 $tf(w,p)$ に，w の類似語 t の値 $tf(t,p)$ を加える．ただし w と t の類似度 $sim(w,t)$ を乗じて加える．これを $new\_tf(w,p)$ とする．$new\_tf(w,p)$ の算出方法を式 (1) に示す．式 (1) の T は，あらすじに単語 w と共起出現する単語の集合である．

$$new\_tf(w,p) = tf(w,p) + \sum_{t \in T} sim(w,t) * tf(t,p) \quad .(1)$$

4.3 単語間の類似度

古くは人が作った類語辞書（Thesaurus）を用いて，類語を見つけることが行われてきた．しかしこの方法では新たに造られた単語の類似語がわからない．そこで，機械的に類似語を算出する方法が考えられてきた．

機械的に単語間の類似度を算出する方法は様々な方法がある．単語を文字列とみて編集距離等の文字列間の距離を

算出し，距離の逆数を類似度とする方法もある．文章内で共起出現する単語を調べ，共起出現回数を類似度とする方法もある．しかしこれらの方法では単語の意味を考慮出来ない．

本研究では単語の意味を考慮するために，単語の分散表現を利用した．単語 w と t の類似度 $sim(w,t)$ は，fatstText が出力した単語の分散表現のコサイン類似度とする．fastText が算出する単語の分散表現（ベクトル）では，意味的に近い単語は近い値のベクトルとなることが多い．十分な文章量を持つコーパスを与えれば近いベクトルが出力されると期待出来る．式 (2) にコサイン類似度の計算式を示す．

$$sim(w,t) = \frac{\sum_i v_{w,i} \cdot v_{t,i}}{\sqrt{\sum_i v_{w,i}^2}\sqrt{\sum_i v_{t,i}^2}} \ . \qquad (2)$$

5. 実験と考察

第**??**章で述べた通り，本研究では小説家になろうの小説メタデータの 2 つを流行語の分析対象とした．この文書集合への実験の適用結果と考察について述べる．収集した小説メタデータのあらすじに対して方式 1 と方式 2 を適用した．流行語の推移粒度の期間 p は 1 ヶ月ごとにした．

5.1 方式 1 の結果

方式 1 を適用した際のトレンドを表 3 に示す．2 つの期間（2010 年 10 月，2019 年 10 月）における出現頻度（4.1 で求めた $tf(w,p)$）が上位の単語 10 個に限定して示し比較する．

表 3 方式 1 の結果

	2010/10		2019/10	
Rank	単語	$tf(w,p)$	単語	$tf(w,p)$
1	の	1066.0	世界	5380.0
2	こと	882.0	の	4371.0
3	世界	736.0	こと	4040.0
4	人	476.0	異	2403.0
5	私	444.0	物語	1678.0
6	それ	440.0	彼	1616.0
7	彼	426.0	それ	1546.0
8	中	376.0	人	1509.0
9	物語	366.0	よう	1402.0
10	少女	362.0	主人公	1402.0

5.2 方式 2 の結果

方式 2 を適用した際のトレンドを表 4 に示す．こちらも 2 つの期間（2010 年 10 月，2019 年 10 月）における出現頻度（4.2 で求めた new_tf）が上位の単語 10 個に限定して示し比較する．

表 4 方式 2 の結果

	2010/10		2019/10	
Rank	単語	new_tf	単語	new_tf
1	青年	2058.2	転移	14877.2
2	こと	1911.7	異	14272.7
3	事	1861.9	別世界	11702.4
4	中	1823.5	世界と日本	11687.4
5	少年	1670.0	世界文化	11066.4
6	同じ星	1638.8	新しい世界	10389.3
7	彼女	1631.1	世界	10291.8
8	辰原	1572.7	不思議な世界	8551.6
9	お付	1569.6	事	8497.1
10	転移	1566.5	現実世界	8449.5

5.3 考察

方式 1，2 共に「の」や「世界と日本」など，名詞ではない単語が含まれている．これらは形態素解析処理の不具合と考えられる．

方式 1 を適用した表 3 を見ると，「の」や「こと」，「それ」といった単語が多く見られる．こういった単語に大きな意味は無いが，日本語では多様される単語である．そのため単純に単語の出現頻度をカウントしただけの方式 1 ではそれらの単語が上位に入る．その結果 2010 年 10 月と 2019 年 10 月とで出現する単語に大きな差が生まれず，流行は掴めない．

方式 2 を適用した表 4 では，表 3 のように「の」等の意味の無い単語が少ない．また 2010 年と 2019 年で出現する単語に大きな変化が見られる．2010 年は 10 位である「転移」が 2019 年では 1 位になっており，期間中に「異世界転生」と呼ばれるジャンルが大きく人気を得たことが分かる．この結果から，方式 2 が流行語の抽出方法として優れていると考えられる．

6. 流行分析ツール

本研究で作成した単語の出現頻度等のデータを利用した流行語分析ツールを作成した．本章では作成したツールについて述べる．

6.1 ツール概要

本ツールのスタート画面を図 6 に示す．本ツールは 2 つの単語と特定の年月を入力とし，それぞれの類似単語と類似度，指定した月での方式 2 における new_tf 値（[0,700] の範囲におさめて対数を取る）をグラフに出力する．指定した単語の類似語がどれだけ出現するのかをグラフで見ることが出来る．また単語同士があらすじの中でどれほど類似しているかを，類似語の観点から確認することが出来る．

図 6　ツールのスタート画面

6.2　Django

本ツールでは Web アプリケーション作成のために Django [12] を利用した．Django は Python で実装された Web アプリケーション作成のためのフレームワークである．2005 年にオープンソースとして公開された．Django の特徴として，Web アプリ作成のための基本的な機能が一通り揃っていることが挙げられる．またログイン機能の実装に必要なユーザー認証等の機能が充実している．さらにデータベース管理画面の自動生成が可能であり，簡単にデータの追加，閲覧，更新，削除が出来る．ライブラリを利用することで機能の拡張も容易である．Django 公式サイトによると Django を使用しているサイトとして，Instagram [13] や National Geographic [14] などがある．

6.3　データフロー

Django を用いた Web アプリケーションのデータフローは図 7 のようになる．Django プロジェクト内の各要素の役割を表に示す．図 7 のデータフローで，受け取ったリクエストからレスポンスを返すまでの流れは以下の通りである．

(1) リクエストを受け取り，ミドルウェアが処理を加える
(2) URL にマッピングされたビューを呼び出す．
(3) クエリを実行し，DB からモデルオブジェクトを取得する．
(4) 取得したモデルオブジェクト等をテンプレートに入れてレスポンスを用意する．
(5) ミドルウェアがレスポンスに処理を加える．
(6) レスポンスを返す．

6.4　ツールを用いた実験

実際に「転生」と「魔法」の 2 単語と，2019 年 10 月を入力としたときのグラフを図 8 に示す．x 軸と y 軸はそれぞれの検索語に対する類似度である．グラフ上の円が左上と右下に分かれて分布しているため，「転生」と「魔法」は

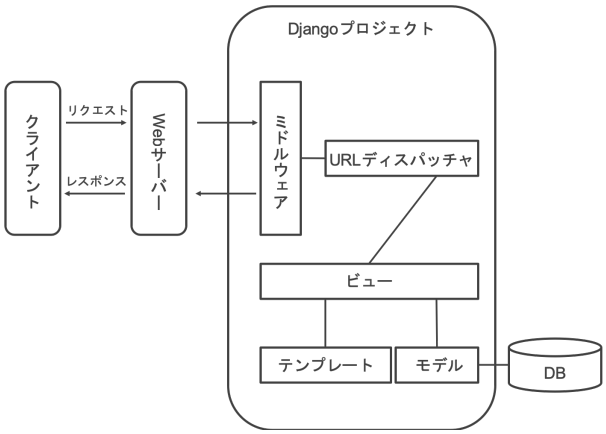

図 7　Django を用いた Web アプリケーションのデータフロー

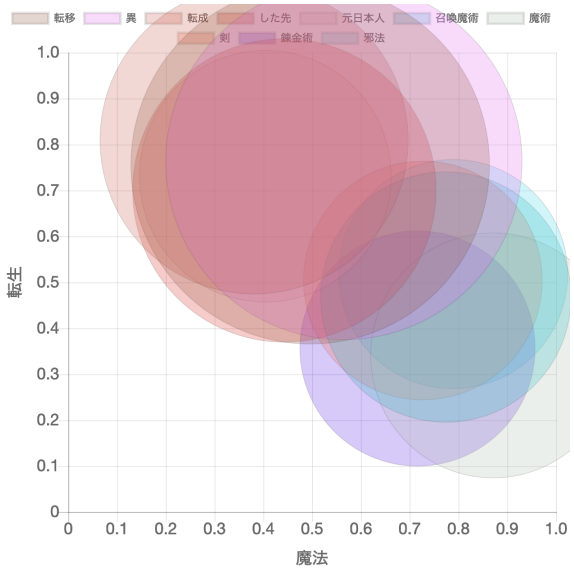

図 8　「転生」,「魔法」を入力した場合のグラフ

あまり似ていないことがわかる．対して「転生」と「転移」を入力とした場合のグラフを図 9 に示す．ここではグラフ上の円が右上に偏っているため，2 つの単語が類似していることがわかる．

7.　関連研究

Google 社が提供する Google Trends [15] では検索語の流行を知ることが出来る．Wikipedia [16] によると公開は 2006 年である．指定した期間での Google 検索における検索語の人気度を折れ線グラフとして表示することが出来る．この人気度は相対的なものであり，指定した期間で最も高い人気度を 100 とする．複数の単語を比較して表示することも出来る．Google Trends ではユーザーが入力する検索語の頻度のみを考慮しており，関連語や類似単語の頻度を考慮していない．

2013 年に中島らはあるトピックに関するブログ記事投稿状況から，そのトピックがメジャーな流行語になり得るかを判定する研究を行った [17]．過去に流行語となったト

表 5　Django プロジェクトの各モジュールの役割

要素	役割
ミドルウェア	リクエストを受け取る際・レスポンスを返す際に処理を加える．
ビュー	レスポンスには Web ページの HTML，リダイレクト等が含まれる．
モデル	モデルとデータベースのテーブルを紐づける．
テンプレート	動的に生成される HTML コンテンツ．
URL ディスパッチャ	URL とビューをマッピングする．

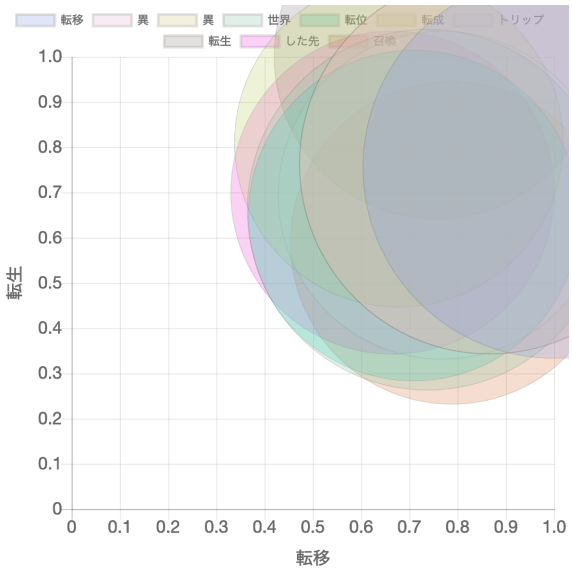

図 9　「転生」,「転移」を入力した場合のグラフ

ピックがブログ上でどのように拡散したかを分析することで，流行語の早期発見を行った．本研究では Yahoo!知恵袋の質問文及び小説家になろうのメタデータ群を対象に分析を行った．今後の分析対象としてブログの投稿記事を利用したい．

SNS から流行を捉える研究が複数行われている．2010 年に白木原らは Twitter から流行を知る研究を行った [18]．流行に敏感なユーザーを検出する方法を提案している．そのために急激に発言数が上昇（バースト）する時間帯を検出するバースト検出アルゴリズムを利用し，バーストした単語に早い時期から反応したユーザーを検出している．白木原らは本研究では考慮していない単語の出現速度を用いている．本研究の今後の課題として単語の出現速度を考慮した分析も行いたい．

8. おわりに

消費者の動向を知り，商機に繋げるために流行語分析は重要である．文書群から流行語を得る方法として，単語の単純な出現頻度カウントや単語の出現速度が知られている．本研究では類似語を考慮した流行語抽出手法を提案した．流行語を抽出する前に，抽出対象の文書群をコーパスとして単語の分散表現を獲得した．特定の単語の出現頻度をカウントする際，獲得した分散表現を基に単語の類似語を導出し，その類似度と出現頻度を掛け合わせた．実際になろう小説 API から取得した小説メタデータのあらすじを対象に提案方式を適用した．その結果，素朴な単語頻度による流行語抽出よりも，より意味を考慮した流行語抽出が出来た．同じ意味を持つ単語でも単純な頻度カウントによる方式 1 では出現傾向にばらつきが見られた．しかし類似語を考慮した方式 2 では，同じ意味を持つ単語の出現傾向のばらつきが見られない．そのため流行語の抽出において，類似単語を考慮することは単純な出現頻度のカウントより優れていると考えられる．「小説家になろう」の小説メタデータを分析する流行語分析ツールも作成した．

今回は単語の分散表現を得るためのコーパスに質問文とを用いた．Wikipedia 等の他のコーパスを用いた場合も比較したい．流行語を抽出する上で 2 つの手法を適用した．単純な出現頻度をカウントしたものと，類似単語の類似度と出現頻度を考慮したものである．今回適用していない手法として，単語の出現速度を導出し，それが急上昇するものを流行語とする手法がある．今後はその手法を取り入れたい．また今回は日本語の「小説家になろう」のメタデータを対象にした．そのため今後は英語の文書群への適用も検討する．

参考文献

[1] 自由国民社:「現代用語の基礎知識」選　ユーキャン新語・流行語大賞，https://www.jiyu.co.jp/singo/ (2020).
[2] 堺雄之介，伊東栄典: オンライン小説の流行語抽出，情報処理学会第 82 回全国大会.
[3] 株式会社ヒナプロジェクト: 小説家になろう - みんなのための小説投稿サイト，https://syosetu.com/ (2020).
[4] 株式会社ヒナプロジェクト: 小説家になろう - Wikipedia，https://ja.wikipedia.org/wiki/%E5%B0%8F%E8%AA%AC%E5%AE%B6%E3%81%AB%E3%81%AA%E3%82%8D%E3%81%86 (2020)
[5] 株式会社ヒナプロジェクト: なろう小説 API - なろうデベロッパー，https://dev.syosetu.com/man/api/ (2020).
[6] SQLite: SQLite Home Page, https://www.sqlite.org/index.html (2020).
[7] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. and Dean, J.: Distributed Representations of Words and Phrases and Their Compositionality, *Proceedings of the 26th International Conference on Neural Information Processing Systems*, NIPS'13, Vol. 2, USA, Curran Associates Inc., pp. 3111–3119 (2013).
[8] Bojanowski, P., Grave, E., Joulin, A. and Mikolov, T.: Enriching Word Vectors with Subword Information,

arXiv preprint arXiv:1607.04606 (2016).

[9] 工藤拓，山本薫，松本裕治：Conditional Random Fields を用いた日本語形態素解析，情報処理学会研究報告自然言語処理（NL），Vol. 2004, No. 47, pp. 89–96（オンライン），入手先 ⟨https://ci.nii.ac.jp/naid/110002911717/⟩ (2004).

[10] Toshinori, S.: Neologism dictionary based on the language resources on the Web for Mecab (2015).

[11] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M. and Duchesnay, E.: Scikit-learn: Machine Learning in Python, *Journal of Machine Learning Research*, Vol. 12, pp. 2825–2830 (2011).

[12] Foundation, D. S.: The Web framework for perfectionists with deadlines — Django, https://www.djangoproject.com/ (2020).

[13] Instagram: Instagram, https://www.instagram.com/ (2020).

[14] Geographic, N.: National Geographic, https://www.nationalgeographic.com/ (2020).

[15] Google: Google トレンド, https://trends.google.co.jp/trends/ (2020).

[16] Wikipedia: Google Trends - Wikipedia, https://en.wikipedia.org/wiki/Google_Trends (2020).

[17] 中島伸介，張建偉，稲垣陽一，中本レン：大規模なブログ記事時系列分析に基づく流行語候補の早期発見手法，情報処理学会論文誌データベース（TOD），Vol. 6, No. 1, pp. 1–15 (2013).

[18] 白木原渉，大石哲也，長谷川隆三，藤田博，越村三幸：Twitter における流行語先取り発言者の検出システムの開発，研究報告データベースシステム（DBS），Vol. 2010, No. 2, pp. 1–8 (2010).