人工知能学会 インタラクティブ
情報アクセスと可視化マイニング研究会(第22回)
SIG-AM-22-01

Twitterのバースト情報に基づく桜の見頃推定

Fine-grained Best Season Estimation for Cherry-blossom Viewing based on Burst Information on Twitter

下園良太[1*]　　乾孝司[1]
Ryota Shimozono[1]　Takashi Inui[1]

[1] 筑波大学大学院
[1] University of Tsukuba

Abstract: SNS services such as Twitter and Facebook have been recognized as useful tools for sightseeing. Tourists usually access to such services to collect local and timely information about their interesting areas. In this paper, we study an automatic estimation method of the best season for cherry-blossom viewing. We applied Kleinberg's burst detection algorithm to tweet data stream and compared with one of the previous estimation methods, the moving average method. From the experimental results, it suggests that the burst-based method can resolve the over-extraction problem with keeping sufficient precision.

1　はじめに

近年，SNS の普及に伴い電子コンテンツが急速に増加している．総務省の日本国内における 1500 人を対象にした調査によると，主要 SNS の利用者は 2012 年の 41.4 %から，2016 年には 71.2 %にまで増加しており，スマートフォンと合わせて SNS の利用が社会に定着してきたことが分かる [1]．これに伴い，SNS で扱われる動画や画像，テキストなど様々な形式のデータが増加している．

SNS として代表的な Twitter[2] ではリアルタイムなコミュニケーションを行うことが出来る．利用者は日常で思ったこと，体験したことをすぐに投稿することができ，利用者はリアルタイムに起こっている「今」について知ることが出来る．

さらに，近年では観光においても SNS などの ICT が活用されている．国土交通省の日本国内における 520 人を対象にした調査では観光者の 9 割以上の人が旅行を計画する際，情報通信機器を利用していることが分かった [3]．その際，観光情報を収集するためにウェブサイトが活用されたり，旅行体験を知るために SNS が利用されている．

観光情報の中でも観光資源の適時性は観光者にとって重要な情報となる．例えば桜や紅葉といった花の見頃や，カニやサンマなど魚介類の食べごろなど，観光資源の「旬」の情報を入手することが出来れば観光計画に役立てることができ，観光をより良いものにすることができる．

現状，旅行ガイドブックにも旬な情報が載っているが，1 ヶ月単位の情報（タラバガニの食べごろは 2 月〜4 月など）となっており旬期間の粒度が粗い問題がある．SNS にはより粒度の細かいリアルタイムな旬の情報があるが，投稿単体では情報不足であったり，過去データからの変遷がみえないなど，情報がまとまっておらず検索などで旬の情報を直接獲得することが難しくなっている．

本研究では観光資源の旬として桜の見頃に着目し，Twitter を用いた日単位の桜の見頃推定手法を検討する．Twitter の位置情報付きの投稿（ツイート）を活用することで各地域における桜の見頃を推定することを目指す．先行研究に移動平均による桜の見頃推定の手法を検討したものがあるが，以下のような課題がある [5]．

- 見頃推定期間が細かく分割される
- 生活周期に合わせて注目度が変化する
- キーワードのみで使用データを選択するとノイズが含まれる
- データ不足

このうち，本研究では「見頃推定期間が細かく分割される」という課題に着目した．これは見頃推定期間がまとまらずに細かくいくつも推定されてしまう問題である．

*連絡先：筑波大学大学院システム情報工学研究科 コンピュータサイエンス専攻
〒 305-0006 茨城県つくば市天王台 1 丁目 1 − 1
E-mail: shimozono@mibel.cs.tsukuba.ac.jp

例えば 3/27 - 4/4 が一つの見頃として推定されるとまとまりがあるが，3/27 - 3/29，3/31 - 4/4 として推定されると見頃推定期間が二つになってしまう．先行研究では見頃推定期間の中から桜に関する事前知識（開花日から満開日となる日数が約 5 日間である．など）を使用して人が直接判断して見頃期間を定めている．

ある期間において，ある事象が急激に増加する現象として「バースト」がある．これは，観光資源がある期間中に見頃となる現象に似ている．本研究では，Twitter のバースト期間を見頃期間として考えて，バースト検知手法を適用しその有効性を検証する．バースト検知手法としては Kleinberg のバースト [4] を用いる．これはバースト状態と平常状態の移り変わりが簡単に行われないため，まとまったバースト期間が現れる．これにより「見頃推定期間が細かく分割される」問題が解決できることを期待する．

本稿ではバースト検知手法を用いて桜の見頃推定を行った結果を報告する．2 章では関連研究，3 章では今回使用したバースト検知手法の説明，4 章では見頃推定とバースト検知の対応づけについて説明を行い，5 章で実験の方法と結果を報告する．最後に，6 章でまとめと今後について述べる．

2 関連研究

本研究の先行研究として，遠藤ら [5] の研究がある．遠藤らは桜や紅葉の見頃推定手法として，各地域での位置情報付きツイートの出現数に対する移動平均を活用した．桜の場合，対象語となる「さくら，桜，サクラ」のツイートの出現数を数え，その出現数の移動平均を一年，7 日間 (一週間)，5 日間 (生物の特性) の三つを定義し，その移動平均の比較を行い見頃推定を行う手法である．前の章で述べたように，この手法には課題点がいくつか存在する．

Kleinberg は，テキストの時系列データである document stream においてバーストを検知できる手法を示した [4]．Kleinberg は，2 種類のバースト解析手法を提案しており，1 つは連続時間で送られるドキュメントデータに対して，バーストしている期間の判定や，バーストの強さ毎のバースト期間の判定ができる．2 つ目は，Enumerating バーストとよばれ，離散時間で送られるドキュメントデータを一つのバッチとして考え，着目するキーワードがバーストしているか否かの判定を行うことができる．

3 Kleinberg のバースト検知

本研究では見頃推定に Kleinberg が提案するバースト検知手法を用いる．これはテキストの時系列データである document stream 内で急激にテキストの発生頻度が上昇しているようなバーストと呼ばれる期間を検知するアルゴリズムである．Kleinberg のバーストには 2 種類あり，時間軸に沿って断続的に発生する document stream の時刻を元にバースト検知を行うもの（連続型）と，単位時間毎に発生した document stream 内の関連文書を数える Enumerating バースト（列挙型）が定義されている．本研究では 1 日単位のテキスト発生に対する見頃推定を行うため，Enumerating バーストを使用する．

Enumerating バーストは，離散時間で送られる文書の集合に対して適用される．本稿では，各日ごとのツイート集合を一つの文書集合の単位とし，以下では単にツイート集合と呼ぶ．2 状態オートマトン $\mathcal{A}^2$ を定義し，2 つの状態を非バースト状態 q_0，バースト状態 q_1 とおく．入力に対して状態が遷移することにより 2 つの状態を切り替える．着目するツイートを「関連ツイート」，そうでないツイートを「非関連ツイート」とし，バーストか否かはツイート集合中の関連ツイートの割合によって決まる．

解析期間において，n 個のツイート集合 $B_1, ..., B_n$ が離散時間で送られてくる状況を考える．t 番目のツイート集合を B_t とし，そのツイート集合に含まれるツイートの数を d_t とおく．さらに，ツイート集合 B_t に含まれる関連ツイートの数を r_t とおく．解析期間における全てのツイートの数 D を $D = \sum_{t=1}^{n} d_t$，全ての関連ツイートの数を $R = \sum_{t=1}^{n} r_t$ と表す．

次に，オートマトンの 2 状態にそれぞれ期待値を割り当てる．初期状態である非バースト状態 q_0 には，解析期間全体から算出した期待値 $p_0 = R/D$ を割り当てる．バースト状態 q_1 には，p_0 にパラメータ s をかけた値である $p_1 = sp_0$ を割り当てる．ただし，$s > 1$ であり，$p_1 \leqq 1$ となるような s でなくてはならない．s はバーストの見なされやすさを表している値であり，この値が小さいほど，ツイート集合中の関連ツイートの割合が低くてもバーストと見なされやすくなる．バースト性の解析は，n 個のツイート集合が与えられたときの状態の系列を通るためのコスト計算によって行う．考えられる状態の系列のうち，最も系列のコストが小さいものが解となり，その系列の状態に応じて，バースト期間と非バースト期間が決定される．状態遷移は d_t と r_t によって定まる．状態の系列は $\boldsymbol{q} = (q_{i1}, ..., q_{in})$ と表され，q_{in} は，n 番目のツイート集合によって決定された状態 $q_i (i = 0, 1)$ である．ツイート集合中の関連ツイートが二項分布 $B(d_t, p_i)$ に従って現れるという考えに基づき，状態 q_i にいることに対してコストを与える関数 $\sigma\ (i, r_t, d_t)$ を以下のように定義する．

$$\sigma\ (i, r_t, d_t) = -\ln[\binom{d_t}{r_t} p_i^{r_t} (1 - p_i)^{d_t - r_t}] \qquad (1)$$

ただし，頻繁に状態遷移が起こると，バースト状態と非バースト状態が頻繁に切り替わることになる．そこで，現在の状態 q_iから次の状態 $q_j(j = 0, 1)$ へ，状態遷移を妨げるためのコスト関数 $\tau\ (i, j)$ を定義する．

$$\tau\ (i,j) = \begin{cases} \gamma\ \ln n & (j > i) \\ 0 & (j \leqq i) \end{cases} \tag{2}$$

τ は，パラメータ γ によって調節されるが，特に理由がない場合は $\gamma\ = 1$ とする．以上に述べた，ある状態 q にいることに対してコストを与える関数 σ と，状態遷移のコスト関数 τ を使って，状態の系列 $\boldsymbol{q}$ を通るためのコスト関数は以下となる．

$$c(\boldsymbol{q}|r_t, d_t) = \sum_{t=0}^{n-1} \tau\ (i_t, i_{t+1}) + \sum_{t=1}^{n} \sigma\ (i_t, r_t, d_t) \tag{3}$$

オートマトン $\mathcal{A}^2$ は二つのパラメータ s，γ によって決まることから，${\mathcal{A}^2}_{s,\ \gamma}$ と表記される．一般的に $s = 2$，$\gamma\ = 1$ と設定するので，本実験ではこれに従い ${\mathcal{A}^2}_{2,1}$ のオートマトンを用いる．

4 見頃推定とバースト検知の対応づけ

本研究では，桜に関するツイートに対して 1 日単位でバースト検知を行なう．そして，バースト検知で検出されたバースト期間を，桜の見頃推定期間とみなす．

5 実験

5.1 実験概要

本実験では見頃推定にバースト検知手法を位置情報付きツイートに適用し，その効果と先行研究の課題の一つである「見頃推定期間が細かく分割される」を改善できるかを確認するため先行研究の手法との比較実験を行った．

気象庁の観測官署において観測される春の植物であり，観光資源の一つである桜を対象とした．実験条件は先行研究と同じ条件として，実験対象地域は「東京都」・「石川県」・「北海道」とし，実際の見頃期間は気象庁の観測データ [6] による桜の開花日から満開日までの期間とした．

文字列「桜」・「さくら」・「サクラ」を含むツイートを関連ツイートとし，地域ごとにバースト検知を行った．実際の見頃期間に対して，先行研究 [5] による移動平均を使った見頃推定期間と Kleinberg のバーストによる見頃推定期間を比較した．

5.2 データセット

データセットは，先行研究で行われたものと同じ条件にしたいため，同じ期間における同じ位置の位置情報付きツイートを使用した．2015 年 2 月 17 日から 12 月 31 日までの期間で，日本の領土を含む範囲である緯度経度が 10 進法表記で 120.0 ≦ 経度 ≦ 154.0 かつ 20.0 ≦ 緯度 ≦ 47.0 の位置情報付きデータとする．データ数は，約 5,000 万件であった．今回対象地域の東京都では約 850 万件，石川県では約 34 万件，北海道では約 190 万件となった．

5.3 実験結果

見頃推定の実験結果を図 1 から図 3 に示す．桜の開花時期の 2 月から 5 月の間における，気象庁の開花日と満開日の観測データ，先行研究の移動平均手法 (AVE, Average）の見頃推定期間と Kleinberg のバースト検知手法（Burst）の見頃推定期間をまとめた．また，正解期間である，開花日から満開日の期間を，図中の黒い横線で表している．

図 1 の東京都の見頃推定結果では，実際の見頃 3/23 - 3/29 に対して，AVE では 3/24 - 4/4 を含む 6 つの見頃期間が推定され，Burst では 3/23 - 4/6 の 1 つの見頃期間が推定された．図 2 の石川県の見頃推定結果では，実際の見頃 3/31 - 4/4 に対して，AVE では 4/1 - 4/9 を含む 6 つの見頃期間が推定され，Burst では 4/1 - 4/14 の 1 つの見頃期間が推定された．図 3 の北海道の見頃推定結果では，実際の見頃 4/22 - 4/26 に対して，AVE では 4/22 - 5/1 を含む 8 つの見頃期間が推定され，Burst では 4/23 - 5/11 を含む 2 つの見頃期間が推定された．

5.4 考察

AVE と Burst を比較すると，どの地域においても AVE による推定期間数よりも Burst による推定期間数が少なくなっており，大きく削減された．これは，Kleinberg のバースト検知の際の，状態遷移コストによるバースト期間の分割を抑える効果のためではないかと考える．推定期間の数や推定期間の長さを考えていく上で，パラメータ s と γ の値を設定することが重要である．パラメータ s と γ はバーストの検出具合に関わるパラメータである．s は状態 q_iであることに対してのコストの調整を行い，γ は状態遷移のコストの調整を行うものである．今後最適な設定をもとめる実験をしていく必要がある．

見頃推定期間の中から最終的に人が見頃期間を定める先行研究に対して，Burst では見頃推定期間の数が大

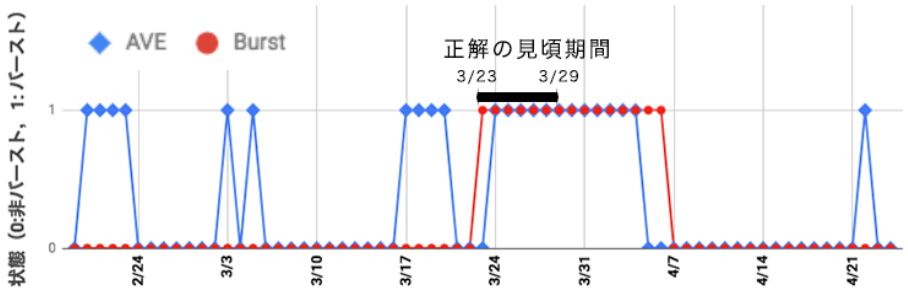

図 1: 東京都の見頃推定結果

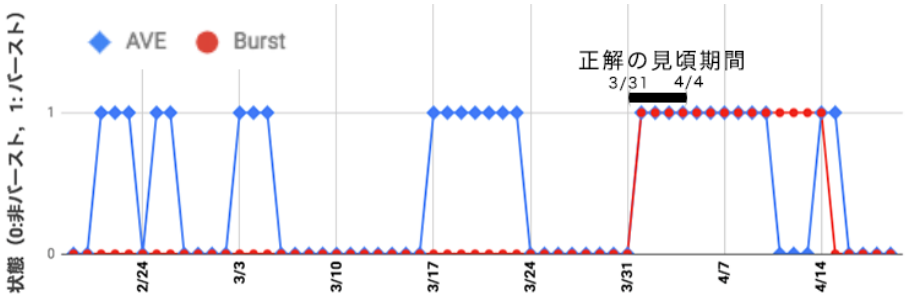

図 2: 石川県の見頃推定結果

きく削減されることから，見頃検出の自動化に近づいていると言える．しかし，図 3（北海道）の Burst の結果を見てみると Burst の見頃推定期間が二つ存在する．図 3 の Burst の 4/1 - 4/3 のツイート内容を確認すると，生物の桜とは関係のない，デジタルコンテンツの桜に関連するイベントについての内容であった．これは意図している対象についての内容ではなく，1 章で述べた課題の一つの「キーワードのみで使用するデータを選択するとノイズが含まれる」に該当する．今後見頃推定の精度を高めるために残りの課題について知見を深め，解決していく必要があると考える．

6　まとめ・今後について

本稿では，観光資源の旬の日単位での推定を行うことを目的として，バースト検知手法を用いた桜の見頃推定を行った．先行研究の移動平均による手法の課題をいくつかあげ，その内の「見頃推定期間が細かく分割される」という問題の解決の期待と，これまでに見頃推定にバースト検知手法が使用されたことが無いことからバースト検知手法の一つである Kleinberg のバースト検知アルゴリズムを適用した．この手法と先行研究の手法の比較実験を行なった．結果，無駄な見頃推定期間が削減され，期待していた問題の解決が出来ることが示唆された．しかし，見頃推定期間が未だ複数出現することから，見頃推定の精度を上げるために，残りの課題を解決する必要があることが分かった．今後はその課題の対応，バースト検知手法のパラメータの調整と，手法の評価を深めるため他の地域や観光資源における見頃推定の実験を行うことを考えている．

謝辞

実験データの収集にあたり，豊橋技術科学大学の吉田光男氏に多大な協力をいただきました．氏に深く感謝いたします．

参考文献

[1] 総務省 平成 29 年情報通信メディアの利用時間と情報行動に関する調査 報告書, http://www.soumu.go.jp/main_content/000564530.pdf, 2018.

[2] Twitter 公式サイト, https://twitter.com.

[3] 観光庁 ICT 活用による観光振興サービスガイド, http://www.mlit.go.jp/common/001080544.pdf, 2014.

[4] J. Kleinberg. Bursty and hierarchical structure in streams. *In Proc. 8th SIGKDD*, pp. 91-101, 2002.

[5] 遠藤雅樹，三富恵佑，佐伯圭介，江原遥，廣田雅春，大野成義，石川博：ツイートを用いた生物季

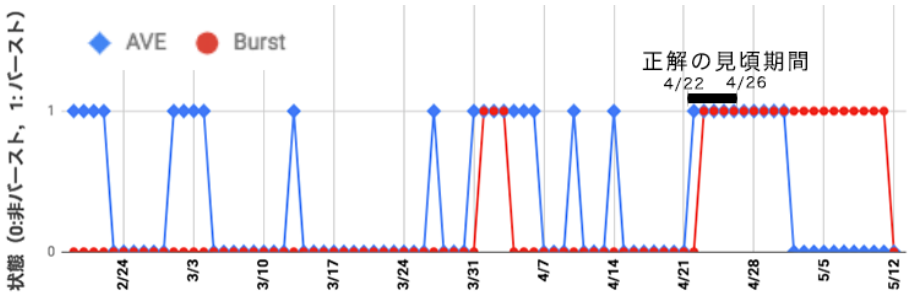

図 3: 北海道の見頃推定結果

節観測の見頃推定手法による情報提供の検討, 観光情報学会誌 12(1), 47-60, 2016.

[6] 気象庁:さくらの観測, 入手先 https://www.data.jma.go.jp/sakura/data/index.html, (閲覧日: 2018/10/02).

深層学習を用いたTwitterからの趣味情報の抽出

Extraction of Interests Information from Twitter Using Deep Learning

若宮 悠希[1*]　砂山 渡[2]　畑中 裕司[2]　小郷原 一智[2]
Yuki WAKAMIYA[1]　Wataru SUNAYAMA[2]　Yuji HATANAKA[2]　Kazunori OGOHARA

[1] 滋賀県立大学大学院工学研究科
[1] Graduate School of Engineering, The University of Shiga Prefecture
[2] 滋賀県立大学工学部
[2] School of Engineering, The University of Shiga Prefecture

Abstract:
In recent years, research to extract personal information from Twitter has been actively conducted. Comments posted on Twitter appear are relatively short and various topics. Therefore comments about a specific interests appear only locally. In this paper, we propose a judgement system whether or not each Twitter user has a designated interest determine after widely learn vocabulary related to specified interests by the deep learning starting from a set of words related to a specific interests.

1　はじめに

近年，インターネットの発展に伴い，Twitter[1]やfacebookといったSNSなどが広く普及することで，オンライン上でのコミュニケーションの場が広く普及してきた．これにより，近しい周囲の人間にとどまらず，距離に縛られない広い範囲の人間と交流を気軽に行うことができるようになった．

これらのサービスを利用するユーザはそれぞれが違う性格や考え方，趣味，嗜好を持つため，交流を行うときは相手の持つ個性が自分のものと合うか，もしくは受け入れられるかどうかが重要となる．そのため，交流相手の個性を十分に理解して受け入れる，もしくは自らが受け入れやすいユーザを選んで交流することが求められる．

交流相手と気が合うかどうかは，同一の話題について興味をもっているかを判断材料とすることができる．Twitterのような不特定多数のユーザが匿名で多様な話題について自由にコメントできるサービスにおいては，検索機能を利用して同じ趣味を持つユーザを選択して交流することもできるが，プロフィールで趣味を明言していないユーザや，サーチワードを含むコメントを投稿したユーザのみしかヒットせず，潜在的に話題に興味を持っているユーザを探すことができないことがあるため，選択の余地が狭まる．

そのため，Twitterに投稿されたコメント集合から，ユーザが特定の趣味を持つか否かを抽出することができれば，交流相手の個性の理解や，新たな交流相手としてユーザを推薦するなど，ユーザ間の交流を手助けが行えることを期待できる．

Twitterから個人情報を抽出する研究は近年盛んに行われるようになってきているが，Twitterにおいては投稿されるコメントが比較的短く，また様々な話題についてのコメントがなされることが多いため，特定の趣味についてのコメントは局所的にしか現れてこない．

そこで本研究では，特定の趣味に関わる単語集合を起点とした深層学習 (Deep Neural Network) により，指定の趣味に関わる関連語彙を潜在的に幅広く学習させた上で，各Twitterユーザが指定の趣味を持つか否かを判定するシステムを提案する．

以下本論文では，2章で関連研究について述べる．3で趣味抽出システムについて述べる．4章で提案システムの有効性の評価について述べ，5章で本論文を締めくくる．

2　関連研究

Twitterに投稿されたコメントから個人情報を抽出する研究は近年盛んに行われてきている．

これまでに，各ユーザの家族構成や所有物，趣味嗜好などを抽出することで個人情報を推定する研究 [1][2]

*連絡先：滋賀県立大学大学院工学研究科 電子システム工学専攻 若宮悠希
〒 522-8533 滋賀県彦根市八坂町 2500
E-mail: of23ywakamiya@ec.usp.ac.jp
[1]https://twitter.com

が行われている．これらの研究では，投稿されたコメント (ツイート) に含まれる単語により個人の情報を抽出する手法を提案しており,「俺のギター」「私の子ども」など，一人称所有格の後に名詞が現れているツイートから，そのユーザの所有物を抽出し,「ギター」であれば「音楽」など所有物が趣味嗜好と関係のあるものならば，所有物を起点にユーザの趣味を抽出する．

また，この他に抽出対象ユーザのプロフィール文やツイートではなく，相互に交流関係のあるユーザのプロフィール文を元に属性を抽出する研究 [3][4] が行われている．この研究では，交流関係のある複数のユーザのプロフィール文に頻繁に出現するものを本人に深く関わりのある単語と仮定して取得することで，これらの単語を元にして対象のユーザに関わりがあると考えられる属性を推定する．

そこで本研究では，自身の学士の研究 [5] を元に，抽出したい趣味を利用者があらかじめ決定して関連単語を与えることにより，趣味と関連する単語として幅広い語彙を網羅した学習を行うことで，推定対象のユーザのツイート集合のみから，指定の趣味を持つか否かの判断を行う．

3 深層学習を用いた趣味抽出システム

3.1 システムの構成

本研究で提案する深層学習を用いた趣味抽出システムの構成を図 1 に示す．

まず，あらかじめ深層学習を用いて構築した各趣味の趣味抽出ネットワークを元に，推定対象の Twitter ユーザのツイート集合に各趣味が現れているかを抽出する．抽出結果を可視化インタフェースを利用して提示することで，システム利用者が結果を解釈する支援とする．

3.2 深層学習による趣味抽出ネットワークの学習

本研究では，推定対象ユーザが指定の趣味に興味を持っているかを判断するために，深層学習による趣味抽出ネットワークを構築，これを元にツイート集合を分類する．

深層学習とは，機械学習の一種で入力と出力の関係をパターンとして学習したネットワークを構築する手法である．構築されたネットワークにより，人間に与えられない細かな分類規則によって新たなデータを分類することができる．

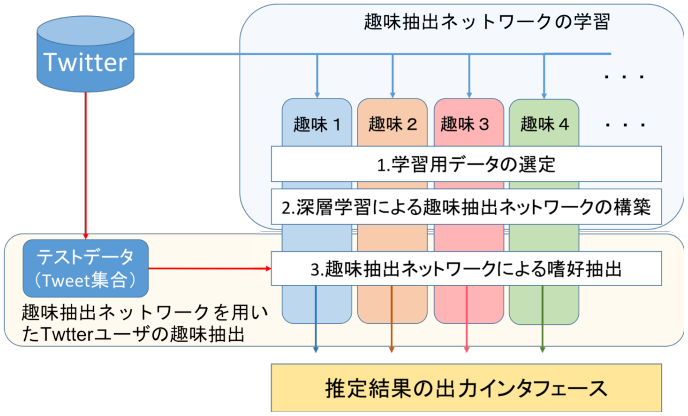

図 1: 趣味抽出システムの構成図

本研究では，深層学習ライブラリの 1 つである Deep Learning for Java(DL4J)[2]を利用して， Deep Neural Network を扱う．

3.2.1 深層学習に利用するデータ集合

本研究では，趣味抽出の対象を Twitter ユーザに定め，各ユーザの投稿したツイート集合を深層学習によって分析し，指定した趣味が文章中に現れているかを抽出する．Twitter ユーザの趣味抽出を行う際は，対象ユーザの投稿した複数のツイートを 1 件ずつ推定していくことで，全体のうち何%に趣味が現れているかを元に最終的な判断をする．

3.2.2 深層学習に利用するデータの収集

深層学習に用いる入力データは，推定対象と同じく，Twitter から収集したツイートを利用する．抽出対象とする趣味が文章に現れているツイートを正例，対象とする趣味が文章に現れていないツイートを負例として正解ラベルをつける．これらのツイート集合を入力データとして，深層学習により趣味抽出ネットワークを構築する．

ここで，ツイートに指定の趣味が現れているか否かは，指定の趣味に関連のある単語集合を設定し，いずれかの単語が文章中に出現しているか否かを判断する基準とする．例えば,「野球」に関していえば「阪神タイガース (プロ野球チーム名)」や「ホームラン (野球用語)」などが考えられる．この単語集合を本研究では「初期単語」と定義する．

Twitter REST API[3]を用いてツイートを収集し，指定の趣味ごとに設定した初期単語を含むツイート，含まないツイートを選別して正例，負例を選択する．

[2]https://deeplearning4j.org
[3]https://developer.twitter.com/en/docs

深層学習の入力データや推定対象の文章は，それぞれ文章から BoW(Bag of Words) に変換して利用する．BoW とは全文章中に出現する単語を並べ，各文章での単語の出現頻度をベクトルで表現したものである．また，抽出対象の趣味特有の表現を学習したいため，使用する単語は 15 ツイート以上に出現した名詞のみとし，その中でも判定に関わらないと考えられる単語はあらかじめ除去する．除去する単語としては,「リツイート」「リプライ」など，Twitter で趣味関係なく広く使われる言葉や，初期単語の有無に関わらずツイートに出現する単語となる．初期単語を含む正例データと，初期単語を含まない約 220 万のツイート集合との間で，出現した単語全てについてカイ 2 乗検定を行い，有意水準 5 %を下回る単語以外を全て除去する．

3.2.3 趣味抽出ネットワークの構築

指定した各趣味に対し，正例，負例の入力データを元に深層学習を行い趣味抽出ネットワークを構築する．入力として BoW を用いることから，入力層ノード数は学習用データ中に出現する名詞の総数となるため，抽出対象となる趣味ごとに入力層ノード数が異なる．その他の深層学習のパラメータは，全て以下のもので統一する．中間層数:3，中間層ノード数:100，中間層活性化関数:Relu，出力層ノード:数 2，出力層活性化関数:Softmax，エポック数:50，ミニバッチサイズ:256，L1 正則化 (係数 0.01)，Adam 利用．

今回，深層学習を行う上で，趣味抽出の結果が後述の評価において最も高い精度となったパラメータを採用した．

3.3 趣味抽出ネットワークを用いた Twitter ユーザの趣味判定

構築した趣味推定ネットワークを利用して対象ユーザのツイートから趣味抽出を行う．趣味抽出を行う際には，推定対象の Twitter ユーザが投稿した複数のツイートの集合を対象とし，ツイート 1 件を 1 データとして，文章中に趣味が現れているかどうかを 2 値で判定する．次に，趣味が強く現れていると正判定されたツイートが，全体のツイート集合に対しどの程度の割合で現れたかを算出してグラフ出力する．抽出結果を提示するための可視化インターフェースとして，統合開発環境である TETDM[6] を利用する．

この時，最終的にユーザが指定の趣味に興味を持っているかどうかは，あらかじめ定めた閾値を超えているかどうかで判断する．指定する趣味の範囲や，どれだけ一般的かにより，文章からの抽出されやすさが異なるため，閾値を一般的な Twitter ユーザの判定結果

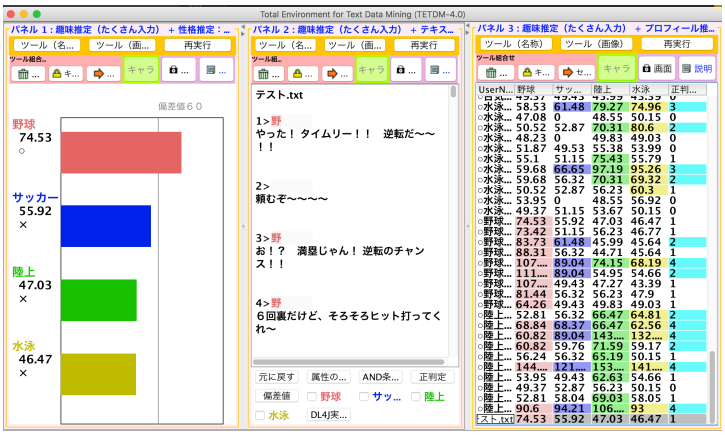

図 2: 趣味抽出システムの使用例

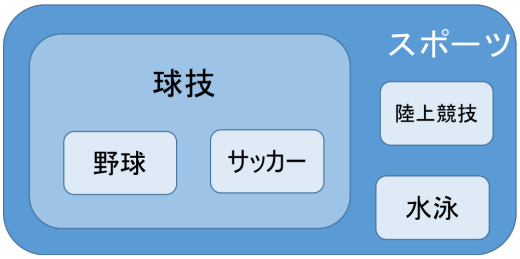

図 3: 趣味の範囲

よりも正判定ツイートの割合が少し高くなるように設定する．ランダムに収集した 15435 ユーザに対し上記の趣味判定を行い，各ユーザの判定結果から正判定ツイート割合の偏差値を算出し，偏差値 60 となる割合を閾値とする．

3.4 趣味抽出システムの使用例

趣味抽出システムの使用例として，Twitter 上での交流相手として，趣味の合うユーザを探し出したい場合，提案システムを利用して，自らの指定した複数の趣味を持つユーザを探し出す例を挙げる．

TETDM を利用した趣味抽出結果の提示例を図 2 に示す．取得した複数ユーザのツイート集合各 100 件程を同時に入力し，趣味抽出システムを起動する．図 2 に示すように，TETDM 上に各ユーザが指定の趣味をどれだけ持つかを正判定ツイート割合の偏差値により表示され，選択したユーザの抽出結果がグラフがツイート集合とともに表示される．図 2 の例では，後述する「野球」「サッカー」「陸上競技」「水泳」の趣味が現れているユーザを，複数入力したユーザの中から抽出しており,「スポーツ」に興味を持つユーザを選択する支援を行える．

3.5 指定の趣味の決め方と抽出方法

本研究では，指定の趣味の関連語彙を学習させることで，対象ユーザのツイート集合に含まれる単語より趣味に興味があるか否かを抽出する．しかし，推定対象となり得る趣味は図 3 に示すようにそれぞれで範囲が異なる．趣味の範囲が広くなる程に内包する属性が多くなるため，初期単語の決定や関連語彙の十分な学習が困難となる．一方で，狭い範囲の趣味を学習する場合，関連語彙を学習しすぎて，少し広い範囲まで抽出してしまう場合がある．例えば,「野球」についての学習を行った結果,「優勝」「攻撃」「応援」「監督」などの関連語彙を網羅する学習を行うが，これらの単語は「スポーツ」全般で利用される言葉である．

そのため本研究では，抽出対象よりも少し狭い複数の趣味を起点として提案手法を用いることで，想定する範囲の趣味を抽出する．

4 趣味抽出システムの評価

3 章で述べた提案手法を評価するため，実際に趣味抽出を行い，抽出可能な趣味の種類や，判定方法などの評価結果を示す．

4.1 提案システムの評価

提案手法を評価するため，一般的な趣味の 1 つである,「スポーツ」趣味を「野球」「サッカー」「陸上競技」「水泳」の各趣味を起点として抽出する．各趣味についてそれぞれ初期単語を設定し，1 つずつ趣味抽出ネットワークを構築し，これらからテストユーザから正しく「スポーツ」趣味を抽出できるかを評価する．

各趣味について設定した初期単語を表 1，それを起点に収集した学習用データ数，入力ノード数 (利用する単語数) を表 2 に示す．また,3.3 章で述べた，各趣味における最終的な正判定の閾値を表 3 に示す．

「スポーツ」評価用として，正解ラベルが正，負のものを合わせて 168 のテストユーザを収集した．収集基準，内訳は以下の通りであり，ユーザの投稿ツイートを読み，受けた印象を元に人手で収集した．

負 特にスポーツに興味がなさそうなユーザ：100

正 各スポーツに興味がありそうなユーザ：68
「野球」9,「サッカー」10,「陸上」10、
「水泳」11,「ゴルフ」10,「卓球」10、
「テニス」5,「ラグビー」2,「合気道」1、
「スケート」1

これらのテストユーザを用いて，各趣味を起点にした場合における「スポーツ」判定を評価した．評価結果として，正判定についての Precision，Recall，F 値を表 4 に示す．

表 4 の結果では，どの趣味を起点として「スポーツ」判定を行っても，高い Precision と低い Recall が目立つことから，提案手法では，誤判定が少ないものの正判定の基準が厳しく設定されていることがわかる．

また，負判定された「正」ユーザの中には，正判定ツイート割合が閾値にわずかに届かなかったものがあった.「入力ツイートの内，正判定ツイート割合が決定した閾値を超えている」ことが最終的な正判定の条件となるため,「スポーツ」に確かに興味があるユーザについても，閾値を下回れば負判定されてしまう．このことから，閾値の設定もさらに慎重に行うべきであることがわかる．

1 つの趣味を起点とした学習により，起点趣味のみに限らず周辺の趣味までの関連語彙を学習し,「スポーツ」の理想的な範囲を網羅できているかを，表 4 の内訳として表 5，6，7，8 により確認した．

「サッカー」起点なら「サッカー」,「野球」起点なら「野球」に興味を持つユーザといったように，起点趣味に興味のあるユーザは高い Recall で判定され，起点趣味に興味を持たないその他のユーザのいくつかを正判定している．ここから，1 つの起点趣味からの学習により，少し広い範囲の関連語彙までを網羅できることがわかる．

一方で，各起点趣味により性判定されやすい趣味が異なる．これは，起点趣味により初期単語が異なることから，学習した語彙が異なることや,「スポーツ」全体に関わる単語を学習できていても，他趣味の固有名詞まで網羅できないことが原因だと考えられる．このことから，起点趣味 1 つのみから「スポーツ」を網羅するのではなく，複数の起点趣味を組み合わせることによって「スポーツ」判定を実現することを考える．

4 つの起点趣味からの学習により構築した趣味抽出ネットワークそれぞれにより趣味抽出を行い，指定した数以上の回数で正判定をされたユーザを，最終的な「スポーツ」趣味を持つユーザであると判断することで，それぞれの起点趣味による学習結果を補い合う．テストユーザを用いた以上の手法の評価結果を，表 9 に示す．

正判定回数を高く設定した場合には，多くの起点趣味で共通して学習した語彙を多く使うユーザが正判定されるなど，判定が更に厳しくなるが，低く設定した場合，それぞれが学習した異なる語彙が判定結果を補い合うことでより適切に判定を行えるようになった．

一方で，どの起点趣味による趣味抽出でも，一度も正判定がつかなかった「正」ラベルのテストユーザが 17 あり，これらのユーザはツイートから受ける印象として，確実に「スポーツ」趣味を持っているのにも関

表 1: 趣味ごとの初期単語

趣味名	初期単語
野球	「ホームラン」「ピッチャー」「バッター」
サッカー	「ゴールキーパー」「フォワード」「ミッドフィールダー」
陸上競技	「ベリーロール」「クラウチング」「背面跳び」「はさみ跳び」「長距離走」「短距離走」
水泳	「クロール」「平泳ぎ」「背泳ぎ」

表 2: 趣味ごとの学習用データ数と入力層ノード数

趣味名	学習用データ数	入力層ノード数
野球	20004(正例:10002)	960
サッカー	20004(正例:10002)	1186
陸上競技	20004(正例:10002)	865
水泳	20004(正例:10002)	1002

表 3: 趣味ごとの閾値 (偏差値 60 となる正判定ツイート割合)

趣味名	閾値
野球	0.131
サッカー	0.081
陸上競技	0.148
水泳	0.185

表 4: 各趣味を起点にした場合における「スポーツ」判定評価結果

趣味名	Precision	Recall	F 値
野球	0.941	0.471	0.627
サッカー	0.935	0.426	0.586
陸上競技	0.911	0.603	0.726
水泳	0.842	0.471	0.604

表 5: 「野球」起点での趣味判定結果の内訳

	「正」予測数	正解ユーザ数	Recall
野球	9	9	1.00
サッカー	6	9	0.67
陸上	5	10	0.50
水泳	0	11	0.00
ゴルフ	4	10	0.40
卓球	5	10	0.50
テニス	1	5	0.20
その他	2	4	0.50

表 6: 「サッカー」起点での趣味判定結果の内訳

	「正」予測数	正解ユーザ数	Recall
野球	3	9	0.33
サッカー	6	9	0.67
陸上	5	10	0.50
水泳	2	11	0.18
ゴルフ	2	10	0.20
卓球	6	10	0.60
テニス	3	5	0.60
その他	2	4	0.50

表 7: 「陸上競技」起点での趣味判定結果の内訳

	「正」予測数	正解ユーザ数	Recall
野球	1	9	0.11
サッカー	4	9	0.44
陸上	9	10	0.90
水泳	5	11	0.45
ゴルフ	7	10	0.70
卓球	8	10	0.80
テニス	3	5	0.60
その他	2	4	0.50

表 8: 「水泳」起点での趣味判定結果の内訳

	「正」予測数	正解ユーザ数	Recall
野球	1	9	0.11
サッカー	4	9	0.44
陸上	5	10	0.50
水泳	4	11	0.36
ゴルフ	7	10	0.70
卓球	7	10	0.70
テニス	1	5	0.20
その他	2	4	0.50

表 9: 複数の起点趣味を組み合わせた「スポーツ」判定評価結果

正判定回数	Precision	Recall	F 値
4	1.000	0.265	0.419
3	1.000	0.368	0.538
2	0.907	0.574	0.703
1	0.839	0.765	0.800

表 10: 判定閾値 52 の場合の複数の起点趣味を組み合わせた「スポーツ」判定評価結果

正判定回数	Precision	Recall	F 値
4	0.975	0.574	0.722
3	0.842	0.706	0.768
2	0.803	0.838	0.820
1	0.626	0.912	0.743

わらず，正判定ツイート割合が閾値に到達していないため，誤判定を受ける結果となっていた，

前述の通り閾値の適切な決定方法を調査するために，正判定ツイート割合が偏差値 40 から 60 まで閾値を変化させて，表 9 の評価を再度行った．この時，最も良い結果を得られた偏差値 52 の結果を表 10 に示す．

閾値を少し下げて適切な値に調整することにより，わずかに閾値に届いていなかった「正」ラベルテストユーザを正しく判定することができた．また，閾値を甘くすることにより,「スポーツ」趣味のないユーザを誤判定してしまうリスクも高まったが，複数の起点趣味で正判定されることを条件に加えることで，Precision を大きく下げることなく，Recall を高くすることができ，より適切な判定結果を得ることができた．これらの結果から，提案手法を用いる際には複数の起点趣味を用意し，それらを組み合わせて趣味抽出を行うべきであると考えられる．

しかし，閾値や組み合わせる起点趣味の数の設定方法などは未だ検討途中であるため,「スポーツ」以外の判定も合わせて行うことで明確なルールを決定する必要がある．また，今回の評価に用いたテストユーザは，個人の主観により収集したものとなるため，複数人での選定等を行い，より信頼性の高いデータを元に評価実験を行う予定である．

5 おわりに

Twitter ユーザを対象として指定の趣味抽出を行い，抽出結果を利用者に提示するシステムの作成を目的として研究を行っている．初期単語と定義した，各趣味に関連の深い語彙を設定することで，これを起点に深層学習により周囲の関連語彙を潜在的に学習することで，対象ユーザの投稿ツイート集合から趣味を抽出して提示するシステムを開発した．

起点となる趣味を元に関連語彙を学習することにより，起点よりも広い範囲の補完を試みるため，一般的な趣味の 1 つである「スポーツ」についての評価を行った．その結果，複数の趣味を起点として組み合わせることにより，学習結果を補い合いより適切な判定結果を得ることができた．

今後の課題として，学習の際に設定する初期単語や起点趣味などの適切決定方法の考察や，テストユーザの増加などにより，更に踏み込んだ評価を行うことで，提案手法の有効性を高めていくことを目標としていきたい．

参考文献

[1] 馬繰美穂, 徳久良子, 寺嶌 立太: ユーザの嗜好と所有物の関係性を用いた属性分析研究報告情報基礎とアクセス技術 2014-IFAT-114,pp.1-6 (2014)

[2] 那須川哲也, 西山莉紗, 金山博, 吉田一星, 大野正樹: 一人称所有格を用いたプロフィール推定, 言語処理学会第 19 回年次大会発表論文集,pp.952-955 (2013)

[3] 上里 和也, 浅井 洋樹, 山名 早人:Personalized PageRank を利用した網羅的 Twitter ユーザ属性推定, DEIM 2016 第 8 回データ工学と情報マネジメントに関するフォーラム D2-2 (2016)

[4] 鈴木 祥平, 池田 拓生, 倉田 陽平, 石川 博:Twitter のユーザプロフィールを用いた観光地の類型化, DEIM 2016 第 8 回データ工学と情報マネジメントに関するフォーラム A2-1 (2016)

[5] 若宮 悠希, 砂山 渡, 畑中 裕司, 小郷原 一智:深層学習を用いた Twitter ユーザの性格推定 2018 年度人工知能学会全国大会 (第 32 回) 3F1-OS-12a-03 (2018)

[6] 砂山渡, 高間康史, 徳永秀和, 串間宗夫, 西村和則, 松下光範, 北村侑也: 統合環境 TETDM を 用いた社会実践, 人工知能学会論文誌 32 巻 1 号,pp.NFC-$A_1 - 12$(2017)

学習済み分散表現への新規単語埋め込みに関する検討

Consideration of new word embedding to learned distributed representation.

森　祥恭[1]　柴田　祐樹[1]　高間　康史[1]

Yoshiyuki Mori[1], Hiroki Shibata[1], and Yasufumi Takama[1]

[1] 首都大学東京大学院システムデザイン研究科

[1] Graduate School of System Design, Tokyo Metropolitan University

Abstract: This paper proposes a method to find vectors of new words on the embedding space without additional learning. While distributed representation is becoming an essential technique for utilizing text information, it has a problem that it cannot calculate embedding vectors of words not included in a corpus. This paper aims to generate the embedding vectors of such words by using the embedding vectors of the words belonging to the same category as the target word. This paper proposes 3 types of calculation methods, and shows the result of preliminary experiments.

1　はじめに

近年，自然言語処理の分野において単語の分散表現を獲得する手法が注目を集めている．単語の分散表現は，テキストコーパスを利用して学習することにより獲得され，コーパス内に存在する単語をベクトルで表現できる．単語の分散表現は機械翻訳[1]やテキストに書かれた感情の分析[2]など多様な用途で利用されている．単語の分散表現を獲得する代表的な手法として，Word2Vec[3]，FastText[4]等が挙げられる．

Word2Vec などの単語の分散表現の特徴として，学習した単語ベクトル間にコサイン類似度を計算することで，単語間の類似度を求めることができる点が挙げられる．また，加法構成性を持つこと，すなわち国と首都や男性と女性のような意味的な関係も単語ベクトル間に関係に反映されていることが挙げられる．しかし，コーパス内に存在しない単語の埋め込みベクトルが得られないことや，特にコーパス内での出現頻度が少ない単語の場合など，学習によって得られたベクトルの信頼性が低いなどの問題点がある[5]. このような問題に対して，コーパスの拡張や subword を用いて対処する研究も行われている[5][6].

上記とは異なるアプローチとして，本稿では，既に学習済みの単語ベクトルからコーパスに存在しない単語のベクトル表現を生成することを最終的な目標とする．学習済みの単語ベクトルを利用して，それらと意味が近い単語ベクトルを生成できれば，大規模コーパスを用いずに語彙数の増加が可能になると考える．その前段階として，生成対象の単語と同カテゴリに存在する単語の埋め込みベクトルなどを利用して，当該単語の埋め込みベクトルを求める手法を提案する. 単一のカテゴリの関係を用いて生成する方法，1 種類の意味的もしくは構文的な関係を用いて生成した単語ベクトルにカテゴリの重心ベクトルを用いて平行移動させる方法，複数のカテゴリ間の関係を用い生成する方法の 3 種類の計算方法を提案し，予備実験を行った結果を示す.

2　関連研究

2.1　分散表現

従来，単語表現として一般的に利用されてきた one-hot 表現は，扱う単語数の分だけ次元を持ち，各要素をそれぞれの単語に割り当てて単語をベクトル表現していた．これに対し，分散表現とは各単語を高次元の実数ベクトルで表現したものであり，テキストコーパスを利用することで求められる．一般に 50～300 次元で表現される．実数のベクトルで表現されることから単語間の類似度を求めることができる．また，加法構成性の性質を持つことから，ベクトル演算を行うことで単語間の意味的な関係を利用することもできる. 例えば，“Paris”-“France”+“Japan”の演算結果が“Tokyo”になるような演算を行える．

2.2　Word2Vec

Word2Vec は，2 層からなるニューラルネットワークを用いて単語の分散表現を学習する手法である．学習する方法として CBoW（Continues Bag-of-Words）モデル[7][8]と Skip-gram モデル[8][9]の二つのモデ

ルが提案されている．CBoW モデルはテキスト内の周辺にある単語を利用して対象となる単語を予測し，単語の分散表現を学習する．Skip-gram モデルは，ある単語が与えられた時に，その周辺の単語を予測し，単語の分散表現を学習する．

2.3 subword による単語ベクトル生成

単語の分散表現はテキストコーパスから獲得するため，1 節で述べた通り，コーパスに存在しない単語や出現頻度の低い単語に対し適切な単語ベクトルが求められないという問題点がある．その解決策の一つとして，単語より小さな単位である subword を用いて分散表現を獲得する手法が提案され，FastText[4]等で利用されている．この手法では，単語の文字 n-gram に対する特徴ベクトルを計算し，それらを足し合わせたものを単語のベクトルとする．例として，“where”という単語を挙げると，文字 n-gram は“wh”，“whe”，“her”，“ere”，“re”となり，それぞれのベクトルを足し合わせることで，“where”の単語ベクトルを生成する．未知語だけでなく略語にも対応できること，品詞の活用形を考慮した単語ベクトルの生成ができることなどの利点が存在する．

3 提案手法

3.1 学習データ

本稿では，Word2Vec で学習した単語の分散表現を利用して，コーパスに存在していない単語のベクトル表現を生成する．Word2Vec の学習には，英語版 Wikipedia のオリジナルのテキストデータに前処理を施した text8[1]を利用する．このデータセットの単語総数は 253,854 である．

英単語のカテゴリ間の関係タイプとして，意味的と構文的の 2 種類を利用する．単語の意味的なカテゴリには英語の概念辞書である WordNet[2]を，構文的なカテゴリに関しては，英語の形態素解析ツールである TreeTagger[3]の品詞情報を利用する．単語の意味的な関係を表 1 に，構文的な関係を表 2 に示す．

表 1. 意味的な関係

カテゴリ 1	カテゴリ 2
国	首都
国	通貨
通貨	首都
男性	女性

表 2. 構文的な関係

カテゴリ 1		カテゴリ 2	
品詞	活用形など	品詞	活用形など
名詞	単数形	名詞	複数形
動詞	単数形	動詞	複数形
動詞	単数形	動詞	過去形
動詞	単数形	動詞	過去分詞
動詞	単数形	動詞	現在分詞
形容詞	原形	形容詞	比較級
形容詞	原形	形容詞	最上級
形容詞	原形	副詞	原形
副詞	原形	副詞	比較級
副詞	原形	副詞	最上級
副詞	原形	名詞	単数形
副詞	原形	名詞	複数形

3.2 単語ベクトルの生成方法

3.1 節で分類した関係タイプを利用して単語ベクトルを生成する．1 つ目の生成手法は，単一のカテゴリの関係を用いて生成する．生成したい単語xの属するカテゴリをc_1，c_1と関係するカテゴリc_2とし，以下の手順により図 1 に示す関係にある単語対の集合Pを求める．

(1) c_2に属し，xと対応関係にある単語yを求める．
(2) c_2に属する単語のうち，y以外で式（1）（2）を満たす単語の集合Yを求める．
(3) Yに属する各単語y'について，c_1に属する単語x'を求め，単語対(x', y')の集合Pを求める．

式（1）（2）におけるd_c，d_l，d_hはパラメータである．求めた$\boldsymbol{x}'$，$\boldsymbol{y}$，Pを利用して$\hat{\boldsymbol{x}}_1$を式（3）により生成する．

$$\cos(\boldsymbol{y}, \boldsymbol{y}') \geq d_c \quad \cdots (1)$$

$$d_l \leq \sqrt{|\boldsymbol{y}' - \boldsymbol{y}|} \leq d_h \quad \cdots (2)$$

$$\hat{\boldsymbol{x}}_1 = \frac{\sum_{(x', y') \in P} (\boldsymbol{x}' - \boldsymbol{y}' + \boldsymbol{y})}{|P|} \quad \cdots (3)$$

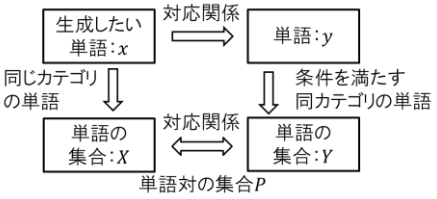

図 1. 単語間の関係図

[1] http://mattmahoney.net/dc/text8.zip -O text8.gz
[2] https://wordnet.princeton.edu/
[3] https://www.cis.uni-muenchen.de/~schmid/tools/TreeTagger/

2 つ目の生成方法は，式（3）で計算した単語ベクトルをカテゴリの重心ベクトルに基づき移動させて生成する．C_1のカテゴリの重心ベクトルを$\boldsymbol{g_1}$，C_2の重心ベクトルを$\boldsymbol{g_2}$として，生成式を式（4）に示す．

$$\hat{\boldsymbol{x}}_2 = \hat{\boldsymbol{x}}_1 + \boldsymbol{g_1} - \boldsymbol{g_2} \quad \cdots (4)$$

3 つ目の生成手法は複数の関係を用いて単語のベクトルを生成する．複数のクラスの平均を求める場合，クラスごとのデータの件数の比率を考慮するマクロ平均と，考慮しないマイクロ平均の 2 種類が存在する．この 2 種類の平均を利用してそれぞれ単語ベクトルを生成する．関係の総数をL，i番目の関係を用いて式（3）より生成したベクトルを$\hat{\boldsymbol{x}}_{1,i}$としたときのマクロ平均での生成式を式（5）に示す．

$$\hat{\boldsymbol{x}}_3 = \sum_{i=1}^{L} \frac{\hat{\boldsymbol{x}}_{1,i}}{L} \quad \cdots (5)$$

i番目の関係についての単語対集合をP_i，xに対応する単語をy_iとしたときのマイクロ平均での生成式を式（6）に示す．

$$\hat{\boldsymbol{x}}_4 = \frac{\sum_{i=1}^{L} \sum_{(x',y') \in P_i} (x' - y' + y_i)}{\sum_{i=1}^{L} |P_i|} \quad \cdots (6)$$

4 予備実験

4.1 実験概要

本稿では予備実験として，単語ベクトル既知の単語について提案手法によりベクトル表現を生成し，既知のベクトルと比較することで評価を行う．評価指標として，コサイン類似度とユークリッド距離を採用する．式（1）（2）のパラメータとして，（d_c，d_l，d_h）=（0.5，0，15）とした．

予備実験で使用する関係，単語数，単語の例を表 3 に示す．表 3 における関係の左側にある単語のカテゴリのみから単語$\boldsymbol{x}$を選んで実験を行う．単語対の数は今回使用した対の総数で，対象単語ごとにベクトル生成に利用される単語対は異なる．

4.2 実験結果

単語の分散表現として，Word2Vec を以下の条件で学習したものを使用した．

- 単語ベクトルの次元数：200
- コーパス内での出現回数が 5 回未満の単語を無視
- 予測する単語の前後 5 単語を学習に使用
- 学習モデル：CBoW

首都，動詞の単数形，副詞のそれぞれの単語ベクトルを生成し，コサイン類似度で評価した結果をそれぞれ表 5，8，11 に，ユークリッド距離で評価した結果をそれぞれ表 6，9，12 に示す．複数の関係がなく単一の関係でのみ単語ベクトルを生成した場合に「なし」としている．各単語における使用した単語対数をそれぞれ表 4，7，10 に示す．単語対は，3.2 節で説明した生成手順に従い，それぞれの$\boldsymbol{x}$に対して選ぶ．

表 3. 実験で使用する関係，単語対の数と例

関係	単語対数	単語対の例
首都-国	32	Tokyo-Japan Beijing-China
首都 -ISO code	17	Tokyo-jpy Bern-chf
単数形-現在分詞(動詞)	73	Sing-Singing Dance-Dancing
単数形-複数形（動詞）	80	Sing-Sings Kiss-Kisses
単数形-過去形（動詞）	116	Sind-Sang Shout-Shouted
単数形-過去分詞(動詞)	39	Sing-Sung Hear-Heard
副詞-形容詞	35	Interestingly-Interesting Obviously-Obvious
副詞-単数形（名詞）	27	Interestingly-Interest Importantly-Importance
副詞-複数形（名詞）	30	Interestingly-Interests Idealy-Ideals
副詞-比較級	10	Happily-happier Fairly-Fairer
副詞-最上級	4	Happily-happiest Sadly-Saddest
男性-女性	39	Man-Woman Boy-Girl
単数形-複数形（名詞）	120	Man-Men Woman-Women
形容詞-比較級	73	Big-Bigger Tall-Taller
形容詞-最上級	22	Big-Biggest Large-Largest

首都に関して，$\hat{\boldsymbol{x}}_1$，$\hat{\boldsymbol{x}}_2$は国のカテゴリのみを利用し，それ以外は ISO code のカテゴリも使用して生成した．表 5，6 より$\hat{\boldsymbol{x}}_2$は$\hat{\boldsymbol{x}}_1$より正解データとの違いが大きくなっていることがわかる．一方，$\hat{\boldsymbol{x}}_3$，$\hat{\boldsymbol{x}}_4$では両方とも類似度が高くなっている．

表 4. 首都の単語対数

x	国	ISO code
Tokyo	12	10
Paris	11	3
Cairo	8	10
Beijing	12	0

表 5. 首都のベクトルを生成した際の
正解データとのコサイン類似度

x	$\hat{\boldsymbol{x}}_1$	$\hat{\boldsymbol{x}}_2$	$\hat{\boldsymbol{x}}_3$	$\hat{\boldsymbol{x}}_4$
Tokyo	0.4679	0.3295	0.6437	0.6069
Paris	0.7014	0.6909	0.7083	0.7169
Cairo	0.6313	0.434	0.6725	0.6748
Beijing	0.5530	0.3802	なし	なし

表 6. 首都のベクトルを生成した際の
正解データとのユークリッド距離

x	$\hat{\boldsymbol{x}}_1$	$\hat{\boldsymbol{x}}_2$	$\hat{\boldsymbol{x}}_3$	$\hat{\boldsymbol{x}}_4$
Tokyo	9.4355	10.4886	5.7595	6.2138
Paris	10.2954	10.431	10.2528	9.9456
Cairo	7.6408	9.5623	4.7759	4.6508
Beijing	9.1188	10.429	なし	なし

動詞の単数形に関して，$\hat{\boldsymbol{x}}_1$，$\hat{\boldsymbol{x}}_2$は動詞の現在分詞のカテゴリのみを利用し，それ以外は複数形，過去形, 過去分詞のカテゴリも使用して生成した. “scan”の$\hat{\boldsymbol{x}}_2$およびユークリッド距離で比較した場合の“train”の$\hat{\boldsymbol{x}}_2$を除いて，$\hat{\boldsymbol{x}}_1$が最も正解データとの違いが大きくなる結果が得られた．最も正解データに近くなる生成方法は$\hat{\boldsymbol{x}}_3$か$\hat{\boldsymbol{x}}_4$のどちらかである.

表 7. 動詞の単数形の単語対数

x	現在分詞	複数形	過去形	過去分詞
Sing	5	30	24	14
Take	5	13	10	25
Train	8	32	22	22
Scan	15	5	16	16

表 8. 動詞の単数形のベクトルを生成した際の
正解データとのコサイン類似度

x	$\hat{\boldsymbol{x}}_1$	$\hat{\boldsymbol{x}}_2$	$\hat{\boldsymbol{x}}_3$	$\hat{\boldsymbol{x}}_4$
Sing	0.6334	0.6836	0.8107	0.8684
Take	0.8184	0.8433	0.9061	0.8802
Train	0.4227	0.4285	0.6738	0.8124
Scan	0.6779	0.6498	0.7503	0.7404

表 9. 動詞の単数形のベクトルを生成した際の
正解データとのユークリッド距離

x	$\hat{\boldsymbol{x}}_1$	$\hat{\boldsymbol{x}}_2$	$\hat{\boldsymbol{x}}_3$	$\hat{\boldsymbol{x}}_4$
Sing	8.6609	8.2400	4.6622	4.2194
Take	9.6607	9.0615	7.2080	8.0084
Train	14.2758	14.5287	8.2631	6.4012
Scan	4.6218	5.0638	3.2080	3.2843

副詞のベクトル$\hat{\boldsymbol{x}}_1$，$\hat{\boldsymbol{x}}_2$は形容詞のカテゴリのみを利用し，それ以外は単数形，複数形，比較級，最上級のカテゴリを使用して生成した．コサイン類似度に関しては$\hat{\boldsymbol{x}}_2$が最も正解データに近くなる結果が得られた．$\hat{\boldsymbol{x}}_3$，$\hat{\boldsymbol{x}}_4$はコサイン類似度で，正解データとの違いが$\hat{\boldsymbol{x}}_1$よりも大きくなる傾向にあることが確認された．首都や動詞の単数形に比べても，正解データとの違いは大きいため，計算に使用する関係の組み合わせなどを変更するなどして，さらに検討を進める必要があると考える.

表 10. 副詞の単語対数

x	形容詞	単数形（名詞）	複数形（名詞）	比較級	最上級
Apparently	7	4	6	0	0
Interestingly	15	8	7	0	0
Possibly	7	10	16	0	0
Happily	8	8	0	10	4

表 11. 副詞のベクトルを生成した際の
正解データとのコサイン類似度

x	$\hat{\boldsymbol{x}}_1$	$\hat{\boldsymbol{x}}_2$	$\hat{\boldsymbol{x}}_3$	$\hat{\boldsymbol{x}}_4$
Apparently	0.4818	0.5185	0.2896	0.3259
Interestingly	0.2896	0.2937	0.0526	0.1461
Possibly	0.2978	0.3110	0.3051	0.2969
Happily	0.4535	0.4664	0.4651	0.4641

表 12. 副詞のベクトルを生成した際の
正解データとのユークリッド距離

x	$\hat{\boldsymbol{x}}_1$	$\hat{\boldsymbol{x}}_2$	$\hat{\boldsymbol{x}}_3$	$\hat{\boldsymbol{x}}_4$
Apparently	8.0499	7.9195	9.2271	8.7082
Interestingly	7.4403	7.4184	6.7586	6.2766
Possibly	11.3965	11.4336	8.7087	8.4084
Happily	5.9463	6.1349	2.5973	2.6462

首都，動詞の単数形，副詞以外のカテゴリについても, 表 3 に示した全ての関係について実験を行い，それらをまとめた結果を表 13 に示す. 対象単語のカテゴリをカテゴリ 1，単一の関係に使用したものをカテゴリ 2 としている．コサイン類似度とユークリ

ッド距離の両方が$\hat{\boldsymbol{x}}_1$より正解データに近くなった場合は○，その中で最も近くなった場合を◎，どちらかのみ近くなった場合は△，両方とも正解データとの違いが$\hat{\boldsymbol{x}}_1$より大きい場合を×としている．

全体的な傾向としては，このことから，3 つ目の生成手法（$\hat{\boldsymbol{x}}_3$，$\hat{\boldsymbol{x}}_4$）が良好であるが，名詞の単数形では$\hat{\boldsymbol{x}}_2$が優れていることなどから，カテゴリによって適切なものを選ぶことで良い結果が得られる可能性が示された．

表 13. 各手法のまとめ

カテゴリ 1	カテゴリ 2	$\hat{\boldsymbol{x}}_2$	$\hat{\boldsymbol{x}}_3$	$\hat{\boldsymbol{x}}_4$
首都	国	×	◎	◎
動詞の現在形	動詞の現在分詞	△	◎	◎
副詞	形容詞	△	△	△
男性	女性	○	○	◎
名詞の単数形	名詞の複数形	◎	×	×
形容詞	比較級	△	◎	△

5 おわりに

本稿では，単語の分散表現を学習する手法である Word2Vec により得られた単語ベクトルを使用して，追加学習なしに新たな単語ベクトルを生成する手法を提案し，予備実験を行った結果を示した．単語間の関係や，カテゴリの重心ベクトルを利用して単語のベクトル表現を生成することにより，正解のデータに近いものを生成できることを示した．また，カテゴリごとに適した計算方法があることも示した．

本稿で示した実験では，コーパス内で出現頻度の比較的高い単語について生成を行ったため，今後は，未知語やコーパス内での出現頻度が低い単語のベクトル表現を生成し，実験を行っていく．また，他の生成手法や，単語ベクトルではなく，文書分類や要約などのタスクレベルでの評価方法などを検討する予定である．

参考文献

[1] K. Cho, D. Bahdanau, F. Bougares, H. Schwenk, Y. Bengio: Learning Phrase Representations Using RNN Encoder-Decoder for Statistical Machine Translation, Proc. of EMNLP, pp. 1724-1734（2014）

[2] D. Tang, F. Wei, N. Yang, M. Zhou, T. Liu, B. Qin: Learning Sentiment-Specific Word Embedding for Twitter Sentiment Classification, ACL2014, pp. 1555-1565 (2013)

[3] Google Code Archive: Word2Vec, https://code.google.com/archive/p/word2vec/ (2013) 最終アクセス 2019 年 6 月 20 日.

[4] A. Joulin, E. Grave, P. Bojanowski, T. Mikolov: Bag of Tricks for Efficient Text Classification, ACL2017, vol.2, pp. 427-431 (2017)

[5] M-H. Luong, R. Socher, C. D. Manning: Better Word Representations with Recursive Neural Networks for Morphology, CoNLL2013, pp. 104-113 (2013)

[6] J. Zhao, S. Mudgal, Y. Liang: Generalizing Word Embeddings Using Bag of Subword, EMNLP2018, pp. 601-606 (2018)

[7] Q. Le, T. Mikolov: Distributed Representations of Sentences and Documents, ICML2014, pp. 1188-1196, (2014)

[8] T. Mikolov, K. Chen, G. Corrado, J. Dean: Efficient Estimation of Word Representations in Vector Space，ICLR Workshop 2013 (2013)

[9] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, J. Dean: Distributed Representations of Words and Phrases and their Compositionality，NIPS2013，pp. 3111-3119 (2013)