# 共起語を用いた同綴異義語を含むtweetの分類手法

菊川　峻輔†　　湯沢　昭夫††　　小林　亜樹†††

† 工学院大学工学部情報通信工学科　〒163–8677 東京都新宿区西新宿 1–24–2
†† 工学院大学大学院工学研究科電気・電子工学専攻　〒163–8677 東京都新宿区西新宿 1–24–2
††† 工学院大学情報学部情報通信工学科　〒163–8677 東京都新宿区西新宿 1–24–2
E-mail: †c515029@ns.kogakuin.ac.jp , ††cm17051@ns.kogakuin.ac.jp, †††aki@cc.kogakuin.ac.jp

**あらまし**　Twitter では，検索する語が同綴異義語による tweet の検索だった場合，目的としない意味の語を含んだ検索結果も返ってきてしまうため，目的とする意味の語による検索結果のみを得ることが求められる．文書分類の研究報告は数多くあげられているが，tweet のような短文での分類の事例はあまり見られない．本研究では同綴異義語が含まれた tweet を共起語を用いて分類する．提案する分類手法は名詞のみを用いた BoW ベクトルのコサイン類似度による分類，Doc2vec を用いたコサイン類似度による分類，SVM を用いた分類の手法を提案する．実際に投稿された tweet を各分類手法にて分類し，各手法の分類結果を示し，比較をした．

**キーワード**　Twitter, SNS, 分類, 同綴異義語

## 1. はじめに

Twitter は tweet と呼ばれる全角 140 文字，半角 280 文字以内のメッセージが世界中で多数投稿されており，その投稿数は 2018 年 6 月 24 日時点で 1 日あたりおよそ 5 億件とされている [1]．Twitter には検索機能が提供されており，任意の単語を用いて検索することで，Twitter ユーザの tweet を取得することができる．この大量の tweet の中には,「同綴異義語」と呼ばれる，表記が同じでありながら単語の意味が異なる単語が存在する．人物の例で言えば,「羽生」という単語があり，この単語には将棋棋士の「羽生善治」や，フィギュアスケート選手の「羽生結弦」などが主に挙げられる．地名の例では,「日本橋」という単語が挙げられ，東京都の「にほんばし」や大阪府の「にっぽんばし」などを含む．

これら同綴異義語を検索対象として tweet の検索を行った場合，この文字が含まれた tweet が検索結果として得られる．しかし,「羽生」という単語で検索を行った場合，将棋棋士の「羽生善治」についての tweet を検索結果として求めていても,「羽生結弦」についての検索結果も得られてしまうといった例が見られる．この問題を回避するための利用者側の行動例としては，検索クエリを増やす方法が挙げられる．将棋棋士の「羽生善治」について検索したい場合,「羽生」という単語に「将棋」を追加すれば，スケート選手の「羽生結弦」についての tweet はほぼ除外できる．しかし,「羽生善治」に関する tweet が必ずしも「将棋」という単語を含んでいるとは限らない．検索クエリを増やした場合，目的とする tweet を得られやすくなるが，検索対象を狭めてしまうことにつながる [2].

そこで，本研究では短文である tweet であっても，同綴異義語を適切に分類できるかを試みる．本論文の構成として，第 2 章で関連研究について述べる．第 3 章で分類方法について紹介する．第 4 章では各分類方法での分類及び結果について述べる．第 5 章では各分類手法における結果に対する考察を行い，第 6 章にて，まとめと今後の課題について述べる．

## 2. 関連研究

文書分類や語義曖昧性の解消を目的とした研究は多数存在する [3–8]．例えば，城光ら [3] は学習対象となる単語の周辺単語を，特定の品詞を持つものや特定の位置に存在するものに限定し，同義語の獲得を目的とする手法を提案した．単語の意味を求めるという点では本研究と似ているが，同義語を獲得するという点が本研究とは異なる．浦田ら [4] は tweet 中に，語義曖昧性のある単語に正しい Wikipedia の記事を抽出できるエンティティリンキングを行っていた．語義曖昧性の含んだ tweet を対象にしている点は本研究と似ているが，本研究はエンティティリンキングではなく tweet の分類のため，この点で先行研究と異なる．Harada ら [5] は「アップル」という単語を多国籍企業である「Apple.inc」と，果物の「りんご」の意味毎に tweet を分類する手法を提案している.「アップル」という同綴異義語が含まれた tweet を分類するという点では本研究と似ているが，この研究では分類をする際に Latent Dirichlet Allocation(LDA) を用いて分類をしていた．Twitter–LDA モデルでは，各ユーザ毎のトピック分布というユーザ固有パラメータを含むため，1tweet を 1 文書として扱うとは言うものの，モデル構築のためには tweet 元ユーザによる十分な分量の tweet を必要とする．これに対し，検索はアドホックに行われることが多く，検索結果の各 tweet の分類を行うためにそれぞれの投稿主の投稿トピック分布を得なければならず，過剰な処理といえる．荒木ら [6] は文書中の多義語を，相互情報量を用いた場合に，多義語を分類を行う上での危険因子的素性と考え，危険因子と判断された素性を除去することで多義性を解消する手法を提案している．この研究は文書中に多義性のある単語が含まれていた場合のテキスト分類をしている点が本研究と似ているが，文書の対象が tweet ではない点が本研究とは異なる．那須川 [7] は文章の位置，文中の各語の依存関係などから単語

の係り受けに着目して，語義曖昧性を解消する手法を提案している．こちらも，単語の曖昧性を解消するという点では本研究と似ているが，この研究は長文を対象にしていたのに対し，本研究は tweet という短文を扱っていることが本研究と異なっている．藤井ら [8] は単語出現頻度と分類カテゴリとの間の $\chi^2$ 統計による重み付け結果に対して，複数のカテゴリにて重要度が高い単語を分類多義語と定義し，これを用いて文書の自動分類をする研究を行っていた．この研究では分類対象とする文書に新聞記事を用いており，こちらの研究に関しても本研究では tweet という短文を扱っていることから対象としている文書が異なり，先行研究との違いを示すことができる．

## 3. 分類方法

本研究では，100 文字程度の短文で 1 文書が構成されている tweet の分類を行うことを目的とした場合に，分類対象となる tweet 集合以外の追加の情報収集を行わない条件下で，代表的な教師あり分類手法がどの程度の分類精度を実現するのかを測定する．この目的は，このような条件下における分類手法間の比較を行い，特性を明らかにすると共に，今後の分類手法開発のための知見を得ることである．

2 章で述べた通り，文書長の短い tweet などの分類では，分類に必要な情報を得るために，複数の tweet を連結するような手法がみられ，代表的な手法 [3–8] では，投稿アカウント毎に前後の tweet を連結することで，文書長を確保する処理が行われている．これは，アカウントは（少なくとも局所時間的には）一定の話題について投稿を続けるという仮定に基づいており，一定の妥当性は認められるが，当然成立しない場合も相当あると考えられる．また，アカウント毎の投稿データの収集には，利用サービスの提供している API に基づく制約などもあり，実用的なアカウント数に関するデータの収集には分単位の時間がかかるため，外部サービスとして実現しようとすることは困難である．

同綴異義語が混在することが問題となる場面の多くは，同単語を含む検索文脈下であると考えられ，これは，現在の検索利用形態を考えると，利用者の思いつきに対応してアドホックに行われていることがほとんどである．統計的分析などのためであれば，時間を掛けた分類処理であっても差し支えないが，このようなアドホックな利用時に適切な同綴異義語の分類を提供できれば，検索機能性の向上につながる．語義に基づく分類は，オントロジーなどを用いた意味解析の研究も進んではきているが，新語略語の類いも多いマイクロブログにどこまで適用できるかについてはさらなる研究を待つ必要がある．そこで，本研究では，逐次的な人による分類を行うことで，漸進的に分類教師データが得られるであろう場面を想定し，小規模の教師データを用いた，教師あり学習で同綴異義語が含まれた tweet を分類することを目的とする．tweet の分類が各分類方法にてどの程度の精度で分類できるかを測る．

用いる手法は Bag–of–Words ベクトルのコサイン類似度による分類 (3.2 節)，Support Vector Machine(SVM) を用いた分類 (3.4 節)，Doc2Vec を用いた分類 (3.3 節) の 3 つの分類方法によって tweet の分類を行う．Bag–of–Words ベクトルのコサイン類似度から tweet のクラス分類をする理由として，コサイン類似度という基礎的な考えでどの程度 tweet を分類できるかを測定する．SVM はクラス分類の研究に使われており，本研究の目的である同綴異義語を含む tweet 分類において，tweet という短文においてはどの程度の分類が可能かを測定する．Doc2Vec は長文であるテキストデータにおいて多く利用され，ユーザが設定する次元数のベクトルに変換することができる．本研究においては，1tweet のみをベクトル化し，それらのベクトル同士の類似度から tweet が分類が可能かどうかを確認する．

### 3.1 準　　備

各分類方法において必要な段階を述べる．最初に，Twitter 社が提供している Streaming API [9] を用いて tweet を取得する．なお，Streaming API を用いて取得される tweet は，つぶやかれた tweet 全体の 1% となっている．取得された tweet 中のリツイート，引用ツイート，文章が完全に一致した tweet(bot 等) は除外し，70 文字以上の tweet を使用した．70 文字以上の tweet を使用した理由として，あまりにも文章が短い tweet は単語数が少ないことから同綴異義語の意味を判別できず，tweet を分類することが困難だと考えたためである．これらの tweet から，学習 tweet 集合とテスト tweet 集合を考える．学習 tweet 集合はあらかじめ人手で分類した tweet である．学習 tweet 集合を $T_{train}$，$c$ を各 tweet がどの単語に属しているかのクラス，$n$ を tweet の件数とし，式 (1) のように定義する．

$$T_{\text{train}} := \{t_i^c \mid i \in \{n\}\} \tag{1}$$

テスト tweet 集合は $T_{train}$ を基に分類する tweet である．テスト tweet 集合を $T_{\text{test}}$，tweet の件数を $k$ とし，式 (2) のように定義する．

$$T_{\text{test}} := \{t_{n+i}^c \mid i \in \{k\}\} \tag{2}$$

$T_{\text{train}}$ 及び $T_{\text{test}}$ を対象に形態素解析を行う．形態素解析器として MeCab [10] を用い，システム辞書には mecab–ipadic–NEologd [11] を使用する．

各分類手法において，形態素解析された $T_{\text{train}}$ 及び $T_{\text{test}}$ の Bag–of–Words(BoW) を用いる．本研究での BoW ベクトルは，文章中に現れた名詞の出現数をベクトルの成分に持つ．名詞総数を $m$ としたとき，名詞の BoW を $\{w_1, \cdots, w_m\}$ とする．任意の tweet $t_i^c$ のベクトルは式 (3) のように考えることができる [12].

$$\begin{aligned} \boldsymbol{t_i^c} &= (v_{i1}, \cdots, v_{ij}, \cdots, v_{im}) \\ v_{ij} &: t_i^c \text{に含まれる } w_j \text{の個数} \\ 1 &\leqq i \leqq n, 1 \leqq j \leqq m \end{aligned} \tag{3}$$

これら $t_i^c$ の BoW ベクトル $\boldsymbol{t_i^c}$ の集合を $\boldsymbol{T_{\text{train}}}$，$\boldsymbol{T_{\text{test}}}$ とする．

また，教師データとして，Yahoo!ニュースのニュース記事を用いた．Yahoo!ニュースのニュース記事を使用した理由として，Twitter のような主に口語的な表現が使われるテキストではなく，ニュース記事のような文語的な表現で表されているテキス

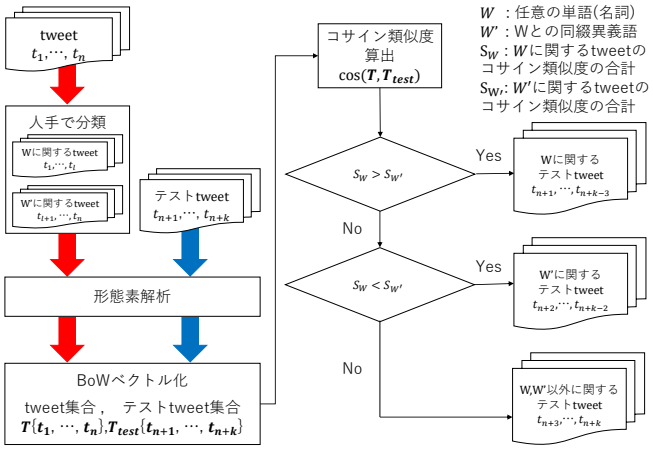


図 1　BoW ベクトルのコサイン類似度による分類

トデータを教師データとした場合での分類結果を比較するために使用する．Yahoo!ニュースのニュース記事は記事タイトルや配信日時，関連記事などは含まず記事本文のみとした．また，記事のページ数が 2 ページ以上のものは 1 ページ目のみを使用した．取得した Yahoo!ニュースのニュース記事の件数を $p$ 件とするとき，ニュース記事集合 $A$ を式 (4) のように定義する．

$$A := \{a_i \mid i \in \{p\}\} \tag{4}$$

取得されたニュース記事は形態素解析をし，BoW ベクトルを作成する．

### 3.2　Bag–of–Words ベクトルのコサイン類似度による分類

本手法は 2 ベクトル間のコサイン類似度を用いて分類する手法である．この分類方法の流れを図 1 に示す．

コサイン類似度は，2 つのベクトルがどの程度近いかを表す．本稿ではコサイン類似度を $\cos(\boldsymbol{x}, \boldsymbol{y})$ と表し，式 (5) のように計算する．ただし，$x_i$，$y_i$ は各ベクトルの第 $i$ 成分を表している．

$$\cos(\boldsymbol{x}, \boldsymbol{y}) = \frac{\boldsymbol{x} \cdot \boldsymbol{y}}{\|\boldsymbol{x}\|\|\boldsymbol{y}\|} = \frac{\sum_{i=1}^{m} x_i y_i}{\sqrt{\sum_{i=1}^{m} {x_i}^2}\sqrt{\sum_{i=1}^{m} {y_i}^2}} \tag{5}$$

ここで，分類のためコサイン類似度に閾値 $\alpha (0 < \alpha < 1)$ を導入する．閾値を導入する理由として，一定の閾値を超える類似度を持つ場合，同じ単語に関する tweet であると判断するためである．任意の単語 W を含む tweet および W の同綴異義語である W' を含む tweet と，学習 tweet 集合 $T_{\text{train}}$ の tweet とのコサイン類似度を算出する．このとき，分類したい tweet $\boldsymbol{t} (\in T_{\text{test}})$ に対して

$$b(c) := \{\boldsymbol{t_i^c} \in \boldsymbol{T_{\text{train}}} \mid \cos(\boldsymbol{t}, \boldsymbol{t_i^c}) \geqq \alpha\} \tag{6}$$

を考える．$b(c)$ は，分類したい tweet $\boldsymbol{t}$ と各単語に関する tweet $t_i^c$ のコサイン類似度が $\alpha$ 以上となる tweet の集合である．次に，$T_{\text{test}}$ に含まれる tweet $\boldsymbol{t}$ のクラスを

$$c(t_i) := \frac{1}{|b(c)|} \arg\max_{c} \sum_{\boldsymbol{t_i^c} \in b(c)} \cos(\boldsymbol{t}, \boldsymbol{t_i^c}) \tag{7}$$

として判定する．各単語において $\alpha$ 以上となるコサイン類似度

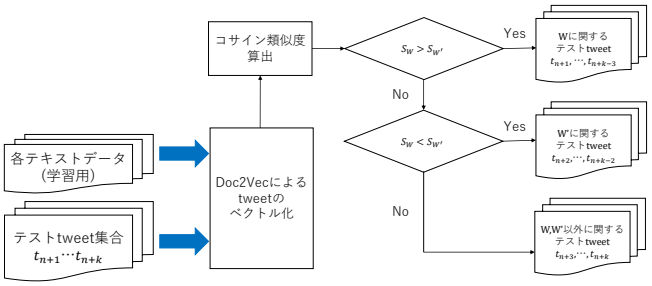


図 2　Doc2Vec を用いた類似度による分類

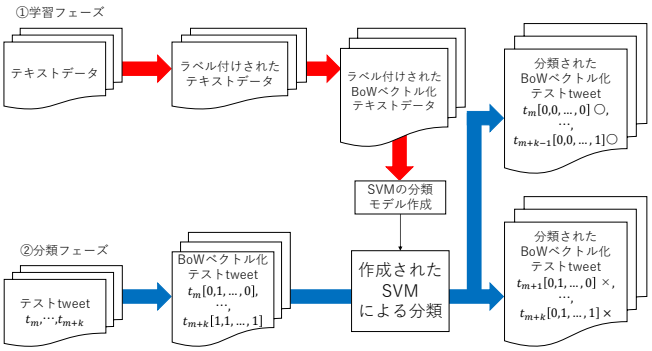


図 3　SVM による分類

の合計値 $S_W$ と $S_{W'}$ を求め，合計値の高い単語の方が tweet $\boldsymbol{t}$ のクラスになる．

本手法において，$S_W = S_{W'}$ となった場合は，tweet のクラスの分類は不可能とした．

### 3.3　Doc2Vec を用いた類似度による分類

Doc2Vec を用いた類似度による分類方法は図 2 のようになる．本手法は gensim の Doc2Vec [15] を用いて教師データのテキストデータとテストデータである tweet をベクトル化し，ベクトル化されたテキストデータと $T_{\text{test}}$ の各 tweet 同士の類似度を求める方法である．ここで言う類似度とはコサイン類似度のことであり，3.2 節の式 (5) のように計算される．また，tweet のクラス判定については，閾値 $\alpha$ を設けた上で，3.2 節の式 (6) 及び式 (7) のように計算する．これらの計算結果より tweet の分類を行う．

### 3.4　Support Vector Machine による分類

Support Vector Machine(SVM) [13] による分類の流れを図 3 に示す．

本手法はテスト tweet 集合 $T_{\text{test}}$ を，SVM を用いて分類する手法である．本手法では 3.2 節で教師データとして利用した学習 tweet 集合 $T_{\text{train}}$ に加え，Yahoo!ニュースから Beautiful-Soup [14] を使用して取得されたニュース記事を教師データとして用いた．図 3 における学習フェーズ，分類フェーズの説明を以下で説明していく．

#### 3.4.1　学習フェーズ

学習フェーズとは，tweet を分類する際に必要とする教師データを SVM に学習させ，分類モデルを作成する段階である．本手法では学習フェーズで，教師データが学習 tweet 集合 $T_{\text{train}}$ の場合，ニュース記事集合 $A$ の場合，ニュース記事集合 $A$ の中から著者が選択したニュース記事集合の計 3 通りについて測

定した．

### 3.4.2 分類フェーズ

分類フェーズは，分類したい tweet 集合を学習フェーズで作成した SVM を用いて分類する段階である．ここで，分類したい tweet に BoW ベクトル化されたテスト tweet 集合 $\boldsymbol{T_{\mathbf{test}}}$ 内の各 tweet を用いる．

## 4. 評価実験

### 4.1 実験目的

本研究の目的であった tweet の分類が，各分類方法においてどの程度の分類精度が得られるかを確認する．

### 4.2 実験条件

同綴異議語を「羽生」「日本橋」「クラウド」とし，これらの語を含む tweet を各分類手法での $T_{\text{train}}$，$T_{\text{test}}$ とした．$T_{\text{train}}$ 及び $T_{\text{test}}$ についての投稿時期や使用した tweet の件数を表 1 に記載する．投稿時期は tweet が投稿された時期を表している．単語はどのような同綴異義語が含まれた tweet を用いたかを表している．件数は各単語において何件の tweet を用いたかを表している．

MeCab のシステム辞書である mecab–ipadic–NEologd は 2018 年 12 月 26 日 18 時 30 分に更新した辞書を使用する．BoW には名詞のみを利用した．

SVM の実装には，scikit–learn の LinearSVC [16] を使用した．カーネルは線形カーネルとし，パラメータは $C = 1.0$ とした．Yahoo!ニュース記事は 2018 年 11 月 19 日時点で「羽生善治」,「羽生結弦」と検索した際の新規 100 件を使用した．また，この中から人手で選択したニュース記事は羽生善治が 67 件，羽生結弦が 64 件となっている．ニュース記事を人手で選択した理由として，各人物の単語が含まれていても，その人物に関する記事ではないものも含まれていたためである．

Doc2Vec を用いて作成されたベクトルの次元数は $N = 300$ とした．対象となる単語 (本研究における同綴異義語) から何個分単語が離れているかを示すウィンドウサイズは window=8 とした．

3 つの分類手法で使用した tweet の統計情報を表 2 に載せる．tweet 数は $T_{\text{train}}$ と $T_{\text{test}}$ の合計を表している．BoW ベクトル次元数は各単語を含む tweet の BoW ベクトルが何次元になっているかを表している．平均単語数は，各単語を含む tweet において何個の単語を含むかを表している．各分類手法では名詞のみを用いていることから，平均単語数は tweet に含まれる名詞数を表している．平均文字数は 1tweet あたりの文字数を表している．ここでの文字数は，名詞以外の品詞も全て含んだ文字数である．

### 4.3 予備実験

予備実験として，$T_{\text{train}}$ を用いて羽生善治に関する tweet 同士，羽生結弦に関する tweet 同士，羽生善治と羽生結弦に関する tweet 同士のコサイン類似度の値が，各閾値以上でどの程度現れるかを算出した．この予備実験の目的は，閾値 $\alpha$ をどの程度の値に設けるかを決定するためである．

この予備実験における実験結果を図 4 に載せる．左の縦軸は，各コサイン類似度の値の範囲における各単語同士の tweet のコサイン類似度の個数とした．右の縦軸は，コサイン類似度の値の範囲における各単語同士の tweet のコサイン類似度の個数の割合を表している．横軸はコサイン類似度の範囲とした．棒グラフは各値における各単語同士の tweet のコサイン類似度の個数を表しており，折れ線グラフは各値における各単語同士の tweet のコサイン類似度の割合を表している．

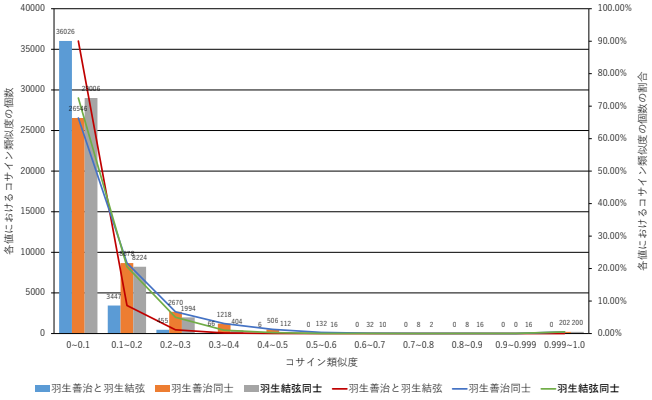


図 4 予備実験結果図

コサイン類似度が 0~0.1 では違う単語同士の tweet の組み合わせの方が数が多い結果となったが，0.1~0.2 以降では，同じ単語に関する tweet 同士の組み合わせの方が数が多くなったため，本研究で用いる閾値 $\alpha$ の値を，0.1 以上 0.2 未満と設定した．この予備実験は単語が「羽生」においてのみの結果であるが，本実験では他の単語においても同等の結果が見込まれると仮定し，各々の単語においても閾値は 0.1 以上 0.2 未満とする．

### 4.4 実験結果

各分類手法において得られた分類結果を以下に載せていく．

#### 4.4.1 BoW ベクトルのコサイン類似度による分類

図 5 は「羽生」という単語が含まれたテスト tweet 集合 $T_{\text{test}}$ を分類する際に要した時間を右の縦軸に秒単位で表し，各閾値における分類精度を左の縦軸に表した図である．横軸はコサイン類似度の閾値を 0.01 刻みで表している．棒グラフは tweet の分類精度を表しており，折れ線グラフは分類をする際に要した時間を表している．また，図 6 は「日本橋」という単語が含まれた tweet をテスト tweet 集合 $T_{\text{test}}$ とした場合の結果であり，図 7 は「クラウド」という単語が含まれた tweet をテスト tweet 集合 $T_{\text{test}}$ とした場合での結果である．

#### 4.4.2 Doc2Vec による類似度を用いた分類

Doc2Vec による分類結果を表 3~ 表 7 に載せる．この実験における類似度の閾値を，$\alpha$=0.1 の場合のみで分類を行い，分類精度を算出した．

#### 4.4.3 SVM による分類

SVM を用いて分類を行った際の分類結果を表 8~12 に示す．正解に対する予測の値をそれぞれ算出した．「羽生」を用いた分類では，$T_{\text{train}}$，$A$，$A$ から人手で選択したニュース記事をそれぞれ教師データとし，「日本橋」,「クラウド」を用いた分類では $T_{\text{train}}$ を教師データとした分類精度の結果をまとめた．テストデータは各単語ともに各単語に関するテスト tweet 集合 $T_{\text{test}}$

表 1　分類にて使用した $T_{\text{train}}$ 及び $T_{\text{test}}$ の条件

| | $T_{train}$ | | | $T_{test}$ | | |
|---|---|---|---|---|---|---|
| 投稿時期 | 10 月 1 日 ~10 月 31 日 | | 12 月 1 日 ~12 月 31 日 | 11 月 1 日 ~11 月 30 日 | | 1 月 1 日 ~1 月 31 日 |
| 単語 | 羽生 | 日本橋 | クラウド | 羽生 | 日本橋 | クラウド |
| 件数 | 善治 200 件<br>結弦 200 件 | 東京 200 件<br>大阪 200 件 | キャラ 180 件<br>用語 180 件 | 善治 50 件<br>結弦 50 件 | 東京 50 件<br>大阪 50 件 | キャラ 50 件<br>用語 50 件 |

表 2　分類で使用した tweet の統計情報

| | 単語 | | |
|---|---|---|---|
| | 羽生 | 日本橋 | クラウド |
| tweet 数 | 500 件 | 500 件 | 460 件 |
| BoW ベクトル次元数 | 3741 次元 | 4893 次元 | 3358 次元 |
| 平均単語数 (名詞のみ) | 22.8 語 | 23.7 語 | 16.0 語 |
| 平均文字数 | 106.0 文字 | 101.6 文字 | 83.7 文字 |

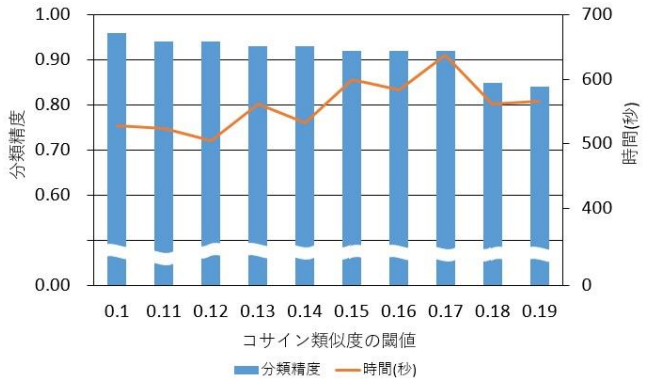


図 5　BoW ベクトルのコサイン類似度による分類方法の結果図 (羽生)

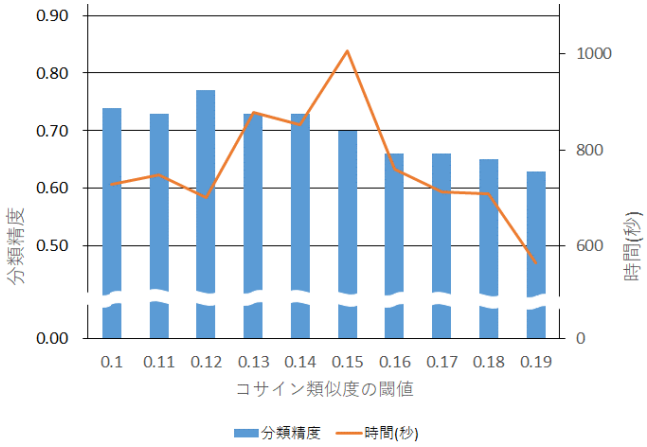


図 6　BoW ベクトルのコサイン類似度による分類方法の結果図 (日本橋)

表 3　教師データが $T_{\text{train}}$ における Doc2Vec による分類精度結果

| | | 予測 | |
|---|---|---|---|
| | | 羽生善治 | 羽生結弦 |
| 正解 | 羽生善治 | 50 | 0 |
| | 羽生結弦 | 46 | 4 |
| 分類精度 | | 0.54 | |

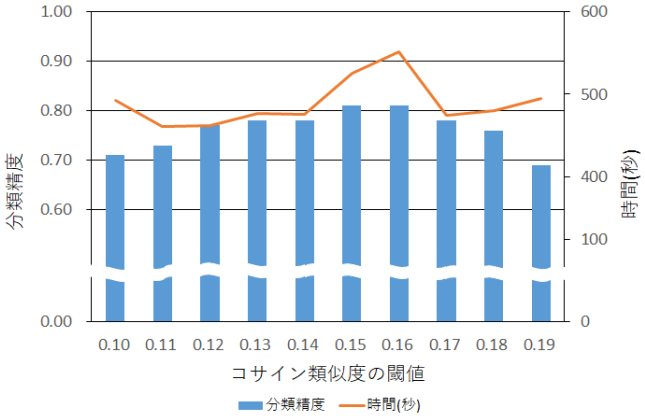


図 7　BoW ベクトルのコサイン類似度による分類方法の結果図 (クラウド)

表 4　教師データが $A$ における Doc2Vec による分類精度結果

| | | 予測 | |
|---|---|---|---|
| | | 羽生善治 | 羽生結弦 |
| 正解 | 羽生善治 | 29 | 21 |
| | 羽生結弦 | 13 | 37 |
| 分類精度 | | 0.66 | |

表 5　教師データが $A$ の中から人手で選択したニュース記事ににおける Doc2Vec による分類精度結果

| | | 予測 | |
|---|---|---|---|
| | | 羽生善治 | 羽生結弦 |
| 正解 | 羽生善治 | 48 | 2 |
| | 羽生結弦 | 49 | 1 |
| 分類精度 | | 0.49 | |

表 6　教師データが $T_{\text{train}}$ における Doc2Vec による分類精度結果

| | | 予測 | |
|---|---|---|---|
| | | にほんばし | にっぽんばし |
| 正解 | にほんばし | 37 | 13 |
| | にっぽんばし | 45 | 5 |
| 分類精度 | | 0.42 | |

表 7　教師データが $T_{\text{train}}$ における Doc2Vec による分類精度結果

| | | 予測 | |
|---|---|---|---|
| | | クラウド (キャラ) | クラウド (用語) |
| 正解 | クラウド (キャラ) | 23 | 27 |
| | クラウド (用語) | 22 | 28 |
| 分類精度 | | 0.51 | |

表 8　教師データが $T_{\text{train}}$ における SVM による分類精度結果

| | | 予測 | |
|---|---|---|---|
| | | 羽生善治 | 羽生結弦 |
| 正解 | 羽生善治 | 28 | 22 |
| | 羽生結弦 | 2 | 48 |
| 分類精度 | | 0.76 | |

表 9　教師データが $A$ における SVM による分類精度結果

| | | 予測 | |
|---|---|---|---|
| | | 羽生善治 | 羽生結弦 |
| 正解 | 羽生善治 | 50 | 0 |
| | 羽生結弦 | 44 | 6 |
| 分類精度 | | 0.56 | |

となっている．予測は SVM にて分類した際の結果を表しており，正解は事前にラベル付けした結果を表している．分類精度は各単語において予測と正解が一致した数の合計値を $T_{\text{test}}$ の数で割った値となっている．

## 5. 考　　察

### 5.1 BoW ベクトルのコサイン類似度による分類方法

BoW ベクトルのコサイン類似度による分類方法が，3 通りの分類方法を通して一番分類精度が高い結果となった．単語が「羽生」の場合は「羽生善治」,「羽生結弦」ともに高い分類精度が得られたが，単語が「日本橋」の場合は分類精度が一番高くても 0.77 程度になった.「羽生」が含まれた tweet の場合,「将棋」「竜王」「スケート」「選手」など，各人物を判別できる単語が tweet された時期に関わらず一定数出現していた．分類対象となる各単語との共起語に明確な違いが表れていたため分類精度が 9 割という高い分類結果になったと考えられる．

一方,「日本橋」という単語に関して,「にっぽんばし」に関する tweet は比較的高い精度で分類できたが,「にほんばし」に関する tweet はあまり高い分類精度を得られることができなかった．これに関して,「にっぽんばし」は主にアニメやゲーム関連の tweet が多くみられた．アニメやゲームに関連する単語は「にほんばし」に関する tweet 中にはあまり含まれていなかったために比較的高い精度で tweet を分類できたと考えられる．しかし,「にほんばし」に関しては 1 つの tweet 文中に地名や店名などの固有名詞が含まれていたが，一般名詞がより多く含まれていたため，人間では「にほんばし」とわかる tweet でも正しい単語の意味に分類できなかったと考えられる．この問題に関して，単語の重みや単語の出現頻度を考慮することで分類精度が大きく変わっていくのではないかと考えられる．

「クラウド」という単語に関しても，ゲームキャラクターである「クラウド」に関する tweet は 9 割を超える精度で分類できたが，インターネット用語である「クラウド」に関する tweet は高い分類精度を得ることができなかった．この結果が得られた要因としては，2018 年 12 月 7 日に「大乱闘スマッシュブラザーズ SPECIAL」というゲームソフトが発売され，このゲーム内に登場する「クラウド」というキャラクターに関する tweet においては，ゲームに登場する別のキャラクターや「クラウド」が使う技の名前などが多く共起し，ゲームキャラクターの「クラウド」に関する tweet 同士では高いコサイン類似度を得ることができたと考えられる．インターネット用語である「クラウド」は主にオンラインストレージとしての意味合いで扱われており，複数サービスにおいて「クラウド」という単語が使われ

表 10　教師データが $A$ の中から人手で選択したニュース記事における SVM による分類精度結果

| | | 予測 | |
|---|---|---|---|
| | | 羽生善治 | 羽生結弦 |
| 正解 | 羽生善治 | 50 | 0 |
| | 羽生結弦 | 32 | 18 |
| 分類精度 | | 0.68 | |

表 11　教師データが $T_{\text{train}}$ における SVM による分類精度結果

| | | 予測 | |
|---|---|---|---|
| | | にほんばし | にっぽんばし |
| 正解 | にほんばし | 38 | 12 |
| | にっぽんばし | 26 | 24 |
| 分類精度 | | 0.62 | |

表 12　教師データが $T_{\text{train}}$ における SVM による分類精度結果

| | | 予測 | |
|---|---|---|---|
| | | クラウド (キャラ) | クラウド (用語) |
| 正解 | クラウド (キャラ) | 50 | 0 |
| | クラウド (用語) | 46 | 4 |
| 分類精度 | | 054 | |

表 13　$A$ の中から関連すると判断した記事を用いて行った確認実験による分類精度結果

| | | 予測 | |
|---|---|---|---|
| | | 羽生善治 | 羽生結弦 |
| 正解 | 羽生善治 | 63 | 1 |
| | 羽生結弦 | 16 | 48 |
| 分類精度 | | 0.87 | |

表 14　教師データが $T_{\text{train}}$ における Doc2Vec による分類精度結果

| | | 予測 | |
|---|---|---|---|
| | | 羽生善治 | 羽生結弦 |
| 正解 | 羽生善治 | 28 | 22 |
| | 羽生結弦 | 24 | 26 |
| 分類精度 | | 0.54 | |

表 15　教師データが $A$ における Doc2Vec による分類精度結果

| | | 予測 | |
|---|---|---|---|
| | | 羽生善治 | 羽生結弦 |
| 正解 | 羽生善治 | 35 | 15 |
| | 羽生結弦 | 39 | 11 |
| 分類精度 | | 0.46 | |

表 16　教師データが $A$ の中から人手で選択したニュース記事ににおける Doc2Vec による分類精度結果

| | | 予測 | |
|---|---|---|---|
| | | 羽生善治 | 羽生結弦 |
| 正解 | 羽生善治 | 45 | 5 |
| | 羽生結弦 | 47 | 3 |
| 分類精度 | | 0.48 | |

ていたことから，インターネット用語の「クラウド」に関する tweet 同士でもあまり同じ単語が見られずコサイン類似度が低くなってしまったと考えられる．

### 5.2　Doc2Vec による類似度を用いた分類

本研究において，各単語を通して最も分類結果が悪かったのがこの分類手法である．Doc2Vec が正しく分類できるかどうかの確認実験を，$T_{\text{train}}$ ではなくニュース記事集合 $A$ の中から各人物に関連していると判断したニュース記事を各人物 64 件ずつ用いて行った．確認実験の結果を表 13 に載せる．

比較的長文なテキストデータであるニュース記事を教師データとし，さらにそのニュース記事をテストデータとして分類した結果，分類精度は 0.87 という結果になった．長文による Doc2Vec の分類は概ね分類精度が高い結果となった．この確認実験により，Doc2Vec による分類は長文データには有効であると考えられるが，tweet という短文のテキストにはあまり有効性がみられないということが，現段階での認識である．この結果に関して，今回表 3〜 表 6 の分類で，テストデータに用いた tweet のベクトルの次元数を 5 次元に設定し，再度調査を行った．その結果を表 14〜18 に載せる．

tweet の次元数を 5 次元と小さくしたが，表 4 と表 15 の比較以外は分類精度の比較の差異が 0.01〜0.09 と大きくは変わらなかった．また，分類精度が高くなったものもあれば分類精度が低くなったものもあり，次元数を減らした方が良くなるということではなかった．次元数を変えても Doc2Vec による実験結果が低いことから，1tweet を 1 文書として類似度を測ることは分類にはあまり適さないと考えられる．だが，Doc2Vec の仕組みについて未だ理解できていない部分はあるため，検討の余地はある．

表 17　教師データが $T_{\text{train}}$ における Doc2Vec による分類精度結果

| | | 予測 | |
|---|---|---|---|
| | | にほんばし | にっぽんばし |
| 正解 | にほんばし | 21 | 29 |
| | にっぽんばし | 20 | 30 |
| 分類精度 | | 0.51 | |

表 18　教師データが $T_{\text{train}}$ における Doc2Vec による分類精度結果

| | | 予測 | |
|---|---|---|---|
| | | クラウド (キャラ) | クラウド (用語) |
| 正解 | クラウド (キャラ) | 23 | 27 |
| | クラウド (用語) | 26 | 24 |
| 分類精度 | | 0.47 | |

### 5.3　SVM による分類

「羽生」に関する tweet の場合，学習 tweet 集合 $T_{\text{train}}$ とニュース記事集合 $A$ の場合で結果が大きく異なった．分類精度が一番高ったのは教師データが $T_{\text{train}}$ となった．教師データが $T_{\text{train}}$ の場合，羽生結弦のみに関する tweet は分類精度が 96% と非常に高い分類精度となった．しかし，教師データが $A$ の場合は羽生善治に関する tweet の分類精度が 100% と異なる人物の方が分類精度が高くなった．このような結果になった原因は，各人物に対する tweet に使われている語彙の違いが原因ではないかと考えられる．

羽生結弦に関する tweet の場合，tweet 内で使われている表現がニュース記事内で使われている表現に大きな差異があることから，教師データを変えた際に分類精度が大きく異なると考えられる．羽生善治に関する tweet の場合，tweet とニュース記事内での語彙が似ていることから分類精度が 100% になったと考えられる．

この問題に関しては，tweet とニュース記事で扱われている語彙の違いを調査することでわかると考えられるが，現段階では確認していない．また，「日本橋」や「クラウド」に関して，教師データが $T_{\text{train}}$ のみということから，判断できる内容が少ないが，正しく分類できなかった tweet には，「日本橋」に関しては東京や大阪を特定させるような単語があまり見られなかったことが分類できなかった原因と考えられる．「クラウド」に関してはインターネット用語における「クラウド」の分類精度が非常に悪かったが，ゲームキャラクターの「クラウド」は 100% 分類できていたことから，目的となる単語の意味を特定できる共起語が多くあることにより，SVM による分類ができるのではないかと考えられる．また，教師あり分類の枠組みであるため，教師データ数と精度の関係を明らかにする必要があると考える．

### 5.4　総　　括

70 文字以上の tweet であった場合，表層的な共起語の出現で

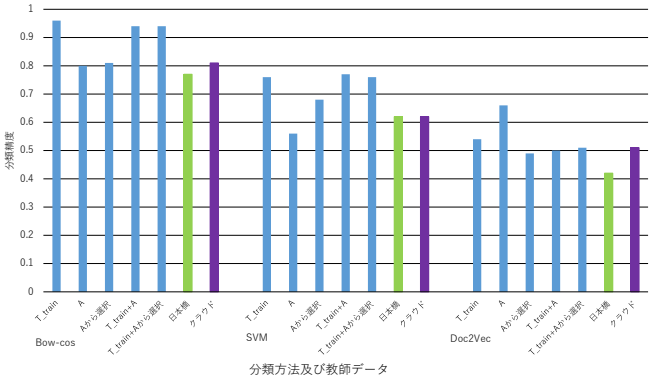


図 8　各分類方法と教師データによる分類精度比較グラフ

は，分類方法にもよるが，ある程度の分類精度を得ることができた．

各単語での実験を通して最も分類精度が良かったのは，BoW ベクトルのコサイン類似度による分類であった．各分類手法及び単語における分類結果の比較図を図 8 に載せる．各棒グラフは教師データにおける分類精度を表しており，青色の棒グラフは単語が「羽生」における各教師データの結果を表している．緑の棒グラフは単語が「日本橋」を含む tweet を教師データとしたグラフであり，紫の棒グラフは単語が「クラウド」を含む tweet を教師データとしたグラフである．比較するにあたって，4.4 節で載せた分類結果以外に教師データが tweet とニュース記事の場合と，tweet と人手で選択したニュース記事の場合の 2 通りを追加し比較をした．

これらの分類を通して分類精度が最も良かったのは BoW ベクトルのコサイン類似度による分類手法であった．コサイン類似度を計算することで，同綴異義語を含む tweet 間で使われている共起語が同じだった場合に，コサイン類似度が高くなることが要因だと考えられる．

SVM や Doc2Vec で分類する手法では，学習データが大きくない場合十分な分類精度を得ることができなかったり，結果が不安定であることが確認できた．教師データの数を増やすといった方法や，両分類手法におけるパラメータを変えることで分類精度は向上できるのではないかと考えられる．

## 6. おわりに

本論文では，同綴異義語が含まれた tweet を，各分類方法によって分類し，どの程度の分類精度が得られるかを検討した．今後の課題として，本論文で示した分類方法以外での分類や，tweet の投稿時期による依存性，及び 70 文字以下という情報量の少ない tweet でも適切に分類できるかという問題，そして本実験で確認された分類精度をより高くするという課題があげられる．

## 文　献


[1] OMNICORE, “Twitter by the Numbers: Stats, Demographics & Fun Facts,” https://www.omnicoreagency.com/twitter-statistics/, 参照 Jan. 2,2019.

[2] 赤部晃一，那須川哲哉，吉川克正，石井甸，“ノイズの海から評判情報を集める，” 電子情報通信学会技術研究報告， 信学技報 NLC2015–46, pp. 13–17, 2016

[3] 城光英彰，松田源立，山口和紀，“文脈限定 Skip–gram による同義語獲得，” 自然言語処理，言語処理学会, Vol. 24, No. 2, pp. 187–204, 2017

[4] 浦田智昭，前田亮，“マイクロブログを対象にしたエンティティリンキングにおける語義曖昧性解消，”DEIM Forum 2018 B3–2

[5] Tomohiko Harada, Kazuhiko Tsuda, “Classifying homographs in Japanese social media texts using a user interest model,” Procedia Computer Science, Vol. 35, pp. 929–936, 2014

[6] 荒木淳，中村文隆，中山雅哉，“多義性を軽減した素性セットによるテキスト分類方式，” 情報処理学会研究報告自然言語処理（NL），Vol. 2003, No. 23, pp. 85–92, Mar. 23, 2003

[7] 那須川哲哉，“文脈情報を利用した自然言語文における構造的曖昧性の解消，” 情報処理学会論文誌，Vol. 36, No. 10, pp. 2362–2370, Oct. 1995

[8] 藤井洋一，鈴木克志，辻秀一，“段落共起情報を利用した文書自動分類方式，” 情報処理学会論文誌, Vol. 42, No. 3, pp. 495–506, Mar. 2001.

[9] “GET statuses/sample,” https://developer.twitter.com/en/docs/tweets/sample-realtime/api-reference/get-statuses-sample 参照 Jan. 9,2019

[10] “MeCab: Yet Another Part–of–Speech and Morphological Analyzer,” http://taku910.github.io/mecab/, 参照 Jan. 9,2019

[11] “mecab-ipadic-NEologd : Neologism dictionary for MeCab,” https://github.com/neologd/mecab-ipadic-neologd, 参照 Jan. 9,2019

[12] 湯浅夏樹，上田徹，外川文雄，“大量文書データ中の単語共起を利用した文書分類，” 情報通信学会論文誌, Vol. 36, No. 8, pp. 1819–1827, Aug., 1995

[13] “Support Vector Machines,” https://scikit-learn.org/stable/modules/svm.html, 参照 Jan. 9,2019

[14] “Beautiful Soup,” https://www.crummy.com/software/BeautifulSoup/, 参照 Jan. 9,2019

[15] “models.doc2vec - Doc2vec paragraph embeddings,” https://radimrehurek.com/gensim/models/doc2vec.html, 参照 Jan. 9,2019

[16] “sklearn.svm.LinearSVC,” https://scikit-learn.org/stable/modules/generated/sklearn.svm.LinearSVC.html, 参照 Jan. 9,2019