DBSJ Japanese Journal
Vol. 14-J, Article No. 15, March 2016

オンラインニュースに関連するツイートのリアルタイムな収集

Real-time Collection of Tweets Relevant to Online News Articles

大西 誠♡ 北川 博之♢

Sei Onishi Hiroyuki Kitagawa

Twitter 等のマイクロブログが注目を集めている．Twitter 上の多くのツイートは，ニュースと関連のある情報を含んでいる．ニュースに関連するツイートを集めることができれば，ニュースに対する人々の意見やニュースの読者の特徴，ニュースの地理的な状況など，ニュースの分析に利用できる．本稿では，ニュースとツイートを内容類似度に基づき照合することで，ニュースに関連するツイートをリアルタイムに収集するための効率的な手法を提案する．まず，ニュースと関連のあるツイートを検出するために，各ニュースに対して内容類似度の閾値を自動的に定める．閾値は，ニュースの投稿よりも前に投稿されたツイートを用いて決定する．また，照合の対象となるニュース集合に対する転置ファイルを用いて，ニュースとツイートの効率的な照合を行う．新しいツイートが投稿されるたびに，提案アルゴリズムである DAAT for variant thresholds を用いて転置ファイルを照合し，内容類似度が閾値を上回るニュースを効率的に検出する．さらに，評価実験により提案手法の有効性を示す．

Microblogs such as Twitter are gaining popularity. A huge number of tweets contain information relevant to news articles. If we can collect tweets relevant to each news article, they can be used for analysis of the news, such as sentiments of people for the news, properties of the news readers, and geo-spatial context of the news. In this paper, we propose an efficient scheme to collect tweets relevant to news articles in real time based on similarity matching of news texts and tweets contents. We systematically determine a similarity threshold for each news article to discover tweets relevant to it. The similarity threshold is decided using a set of tweets posted before the publication of the news article. An inverted file indexing a set of target news articles is used for efficient matching of tweets and the news set in real time. Whenever a new tweet is posted, the inverted file is checked, and news articles whose similarities are above the given thresholds are identified efficiently using our proposed algorithm "DAAT for variant thresholds". Experimental results suggest effectiveness of the proposal.

1. はじめに

マイクロブログの普及に伴い，一般の人々が Web 上に情報を発信する機会が増加した．特に Twitter と呼ばれるマイクロブログでは 1 日平均 5 億件ほどの投稿が行われており [1]，Web 上の重要な情報源として注目されている．Twitter はツイートとよばれる 140 字以内の短いメッセージを投稿し，ユーザ同士で共有することができるサービスである．ユーザはチャットのような感覚で気軽にメッセージを発信できるため，身近な出来事の報告や様々な話題に関する意見・感想など，多種多様な情報を投稿している．

Twitter で投稿されるツイートの中には，ニュースの話題と関連のあるツイートが多く存在していることがわかっている [1]．例えば，政治・社会に対する意見や商品に対する感想など，個人の主観的な評価情報を含む投稿が存在している．このようなツイートのメッセージは，ニュースに対する感情分析や意見抽出の情報源として利用できる [2]．また，ツイートにはメッセージの他にユーザのプロフィール情報や位置情報などが付加されている．このような付加情報を用いることで，ニュースと関連のあるツイートを行ったユーザの年齢や性別，位置情報を取得できる．

また，Twitter は即時性の高いマイクロブログであることが知られている [3][4][5]．そのため，ニュースと関連のあるツイートをリアルタイムに収集することにより，ニュースに対するリアルタイムな分析が可能となる．これにより，地震や台風などの迅速な対応が必要なニュースにおいても，ユーザの意見や位置情報などの有用な情報を取得できる．

そこで本研究では，ニュースと関連のあるツイートをリアルタイムに収集するための手法を提案する．具体的なアプローチとして，ニュースが配信された時刻よりも過去のツイートの傾向を用いて，ニュース毎に類似スコアの閾値を定める．その後，ニュースが配信された後に投稿されるツイートに対し，ニュースとの類似スコアを計算し，閾値以上の類似スコアを持つツイートを収集する．

本論文の貢献を以下に列挙する．

1. ニュース毎に適切な類似スコアの閾値を自動的に定める手法を提案する．
2. 類似スコアの計算，及び閾値との比較を効率的に行う手法を提案する．

本稿の構成は以下の通りである．まず，2 節で関連研究を概観し，3 節で，本研究で用いる類似度指標について述べる．また，4 節で転置インデックスを利用した，既存手法の説明を行し，5 節で提案手法について述べる．最後に，6 節で提案手法の有効性を評価する実験について述べ，7 節で本稿をまとめる．

2. 関連研究

本章では，ニュースと関連のあるツイートを収集するための研究について概観する [6][7][8][9].

Qiu ら [8] は,「リンク付きツイート」を用いてニュースと関連のあるツイートを収集する手法を提案した.「リンク付きツイート」とは，ニュースサイトやブログ等へのリンクが含まれているツイートである．この手法は，リンク先の単語を用いて重要な単語に重みを付けることによって，リンクを持っていないツイートとニュースとの関連付けの精度を向上させるものである．しかし，予めリンク先の本文を人手で抽出しておく必要があり，リアルタイムな処理に対応できないという問題点がある．

Shraer ら [9] は，各ニュースとツイートとの類似スコアをリアルタイムに計算し，ニュース毎に現時点で類似スコアが高い上位 k 件のツイートを常に保持する手法を提案した．しかし，ニュース毎に関連のあるツイートの数は異なることが考えられる．例えば，人気があるニュースの場合だと関連のあるツイートの数は多く，人気がないニュースだと関連のあるツイートの数は少ないことが考えられる．そのため，この手法では関連のあるツイートを収集し損ねるニュースや関連のないツイートを収集しすぎるニュースが存在することになる．

♡ 学生会員　筑波大学システム情報工学研究科
sei0024@kde.cs.tsukuba.ac.jp
♢ 正会員　筑波大学システム情報系情報工学域
kitagawa@cs.tsukuba.ac.jp

[1] https://biz.twitter.com/ja/whos-twitter

また，一般的な情報検索のアプローチとして，全てのニュースに対して一律の閾値を定め，閾値以上の類似スコアを持つツイートを収集する手法が考えられる．しかし，調査を行った結果，ニュース毎に適切な閾値が異なることがわかった．

そこで，本研究ではニュース毎に適切な閾値を定め，閾値以上の類似スコアを持つツイートをリアルタイムに収集することを考える．

3. 類似度指標

本節ではニュースとツイートの関連度を測るために用いる類似度指標について説明を行う．一般的に，同じ単語を多く含む文書ほど関連度は高いと考えられる．そのため，本研究ではニュースの本文とツイートのメッセージに含まれる単語から文書間類似度を計算し，文書間類似度が高いペアほど関連度が高いと考える．本研究では，より精度の高い類似度指標を用いるために複数の類似度指標を用いて精度の比較を行った．まず，3.1 節において，類似度指標を決定するために行った予備実験の内容について述べる．また，3.2 節において，予備実験に用いたデータセットについて説明を行う．最後に，3.3 節で予備実験の結果と提案手法に用いる類似度指標について述べる．

3.1 予備実験

提案手法で用いる類似度指標を決定するために以下の 4 つの類似度指標の精度比較を行った．

- **コサイン類似度：**コサイン類似度は情報検索における伝統的な類似度指標の一つである．精度比較において，以下のような式を用いて計算を行った

$$score(s,u) = \frac{\vec{u} \cdot \vec{s}}{\sqrt{|\vec{u}| \cdot |\vec{s}|}}$$

  $\vec{s}$ はニュースに含まれる単語のベクトルであり，$\vec{u}$ はツイートに含まれる単語のベクトルである．
- **ジャカード係数：**コサイン類似度と同様に情報検索における伝統的な類似度指標の一つである．精度比較において，以下のような式を用いて計算を行った

$$score(s,u) = \frac{|u \cap s|}{|u \cup s|}$$

  $u \cap s$ はニュースとツイートに含まれる単語の積集合であり，$u \cup s$ はニュースとツイートに含まれる単語の和集合である．
- **Content-Based Score：**Content-Based Score は Shraer ら [9] がニュースとツイート間の類似性を測るために用いた類似度指標である．この類似度指標は，Lucene[2] で用いられている類似スコアを拡張したものであり，式は以下のようになる．

$$score(s,u) = \sum_i u_i \cdot idf^2(i) \cdot \sqrt{\frac{s_i}{|s|}}$$

  i は単語を表しており，u_i はツイートに含まれる単語 i の出現回数，s_i はニュースに含まれる単語 i の出現回数を表している．また，$|s|$ はニュースに含まれる全単語数であり，idf はニュースにおける逆文書頻度を表している．idf は以下の式によって計算する．

$$idf(i) = 1 + \log\left(\frac{|S|}{1 + |\{s \in S | s_i > 0\}|}\right)$$

  $|S|$ は全ニュース数を表しており，$|\{s \in S | s_i > 0\}|$ は単語 i を含むニュースの数を表している．
- **Extended Content-Based Score：**Extended Content-Based Score は Content-Based Score を一部拡張したものである．式は以下のようになる．

$$score(s,u) = \sum_i I(u_i) \cdot idf^2(i) \cdot \sqrt{\frac{s_i}{|s|}}$$

$$I(u_i) = \begin{cases} 1 (i \in u) \\ 0 (i \notin u) \end{cases}$$

  $I(u_i)$ はツイート内に単語 i が含まれていた場合は 1，含まれていなかった場合は 0 をとる関数である．

上記の 4 つの類似度指標を用いて精度比較を行った．精度比較は以下の手順で行う．

1. 各類似度指標を用いてニュースとツイートとの類似スコアを計算
2. ニュース毎に類似スコアの高いツイート上位 100 件を取得
3. ニュース毎に MRR(Mean Reciprocal Rank) を用いて評価
4. MRR の平均を計算し，手法毎に比較

MRR とは 1 つの正解について正解が出現した順位を評価する指標である．式は以下のようになる．

$$RR_i = \frac{1}{r}$$

$$MRR = \frac{1}{N} \sum_i^N RR_i$$

r は正解が出現した順位であり，N は取得する件数である．

また，予備実験では Mecab を使って形態素解析を行っており，品詞が名詞の単語を計算の対象として用いる．また，名詞におけるストップワードを取り除くため，品詞細分類が一般，固有名詞，形容動詞語幹，サ変接続の単語のみを計算の対象とした．

3.2 データセット

3.1 節の予備実験で用いるデータセットとして以下の 4 種類のデータを用いる．

- **ニュースデータ：**2014 年 6 月 9 日に Yahoo!ニュースの速報ニュースで配信されたニュース 1037 件の内，ランダムに選択したニュース 30 件を用いる．
- **ツイートデータ：**各ニュースが配信された時刻から 24 時間以内に投稿された日本語のツイートを用いる．
- **IDF データ：**2014 年 5 月 1 日の 0 時 0 分から 6 月 1 日の 0 時 0 分までに Yahoo!ニュースの速報ニュースで配信されたニュース 32295 件を用いる．
- **正解データ：**各ニュースに対して Extended Content-Based Score を用いてランキングを行い，上位 1000 件のツイートとの関連性の評価を行った．ニュースとツイートのペアに対して，評価者が以下の 4 つの中から評価を行った．
  1. 関連がある
  2. やや関連がある
  3. ほとんど関連がない
  4. 関連がない

  3 名の評価者の内，過半数が「関連がある」または「やや関連がある」を選んだツイートをニュースと関連があるツイートと考え，正解データとした．

[2] lucene.apache.org

3.3 結果

予備実験の結果を表 1 に示す．比較した類似度指標の中では，Extended Content-Based Score が最も良い結果を示した．コサイン類似度はベクトルを用いて類似度を計算しているため，単語数が少ないツイートの評価が高くなり，関連のないツイートが上位になる傾向が見られた．また，Content-Based Score は同じ単語を複数個含むようなツイートの評価が高くなり，関連のないツイートが上位になる傾向が見られた．それに対し，Extended Content-Based Score は同じ単語を複数個含むようなツイートの場合であっても，評価が上がらないため Content-Based Score よりも良い結果を示した．また，Extended Content-Based Score がジャカード係数よりも良い結果を示しているのは，*idf* が有効に働いているためだと考えられる．上記の結果より，本手法では Extended Content-Based Score を類似スコアとして用いる．

表 1: 類似度指標の比較結果

cosine	Jaccard	CBS	ECBS
0.0221	0.0289	0.0182	0.0324

4. 既存の転置インデックス照合手法

提案手法では類似スコアの計算及び閾値との比較を行うために転置インデックスを用いた処理を行う．そのため本節では，転置インデックスを用いた既存の照合手法について説明を行う．まず，4.1 節において基本的な手法である DAAT for pub-sub の説明を行う．次に，4.2 節において DAAT for pub-sub を応用した，DAAT for pub-sub with sikipping について述べる．

4.1 DAAT for pub-sub

大規模な文書集合に対するクエリの評価手法として，FLOOD ら [10] によって Document at a time (DAAT) が提案された．DAAT は転置インデックスを利用して，文書毎にクエリの評価を行う手法である．Shraer ら [9] は，DAAT を用いて文書毎にクエリの評価を行い，各文書に対して上位 k 件以内に該当するかを判定する手法である DAAT for pub-sub を提案した．

まず，DAAT for pub-sub で用いる転置インデックスの構造について説明を行う．各ポスティングはニュース ID と部分スコアによって構成されている．部分スコアとは類似度指標の内，単語 i とニュースのみを使って計算することのできるスコアである．例えば，Extended Content-Based Score における部分スコアは以下のようになる．

$$ps(s,i) = idf^2(i) \cdot \sqrt{\frac{s_i}{|s|}}$$

部分スコアの定義により，同じニュース ID をもつポスティングの部分スコアを用いて $I(u_i) \cdot ps(s,i)$ を計算し，それらを合計することによってニュースに対する実際の類似スコアの計算を行うことができる．

次に，DAAT for pub-sub による類似スコアの計算方法について説明する．DAAT for pub-sub のアルゴリズムを Algorithm1 に示す．各ポスティングリスト内のポスティングはニュース ID 順に並んでいるものとする．まず，各転置リストの先頭のポスティングをカレントポジションとする．カレントポジション内において最もニュース ID が低いニュースに対して $I(u_i) \cdot ps(s,i)$ を計算し，それらを合計することによって類似スコアを計算する．類似スコアを計算した後は，計算を行ったニュース ID の k 番目のツイートが持っている類似スコアと比較を行う．k 番目のツイートが持つ類似スコアよりも新しい類似ツイートの持つスコアが大きい場合は，k 番目のツイートを破棄し，新しいツイートを保持する．ポスティングに対して類似スコアの計算を行った後，その転置リストのカレントポジションを次のポスティングに移動する．全てのポスティングに対して計算を行ったら処理を終了する．結果的に，ニュース ID が小さいニュースから順に類似スコアを計算及び閾値と比較を行うことができる．

Algorithm 1: DAAT for pub-sub([9] より引用)

Input : 転置インデックス S
Input : ツイート u
Input : $R_{s_1}, R_{s_2}, ..., R_{s_n}$ - ニュース毎に保持しているサイズ k のミンヒープ
Output: 更新されたミンヒープ $R_{s_1}, R_{s_2}, ..., R_{s_n}$

```
ツイート u に含まれる単語 i の転置リスト L_1, L_2, ..., L_n
for i ∈ [1, 2, ..., |u|] do
    転置リスト L_i の先頭をカレントポジションとする
while 計算を行ってないポスティングがある do
    s ← min_{1≤i≤|u|} L_i.cur
    // cur = カレントポジションのニュース ID
    score ← 0 for i ∈ [1, 2, ..., |u|] do
        if L_i.cur = s then
            score ← score + u_i · L_i.curPs
            // curPs = カレントポジションの部分スコア
            L_i のカレントポジションを 1 つ進める
    μ_s ← min. もし |R_s| = k なら R_s における最小のスコア,
    そうでなければ 0
    if μ_s < score then
        if |R_s| = k then
            R_s における最小のスコアのツイートを削除
        R_s に (u, score) を追加
return R_{s_1}, R_{s_2}, ..., R_{s_n}
```

4.2 DAAT for pub-sub with skipping

DAAT のアルゴリズムを応用することによってツイートの評価を効率的に行う手法が提案されている．有名なものとしては，Broder らが提案した WAND[11] が挙げられる．これは，ニュースとツイート間の内容類似度の上限値を用いることによって，計算対象のニュースを減らす手法である．また，shraer ら [9] は，WAND を DAAT for pub-sub に適用した DAAT for pub-sub with skipping を提案した．これは，平衡二分木を用いることによって計算対象のニュースを高速に見つけ出すものである．以下では DAAT for pub-sub with skipping について説明を行う．

DAAT for pub-sub with skipping では，ニュースとツイート間の内容類似度の上限値を計算することによって，それ以上の閾値をもつニュースとの計算を削減する．転置リストを用いることによって単語 i が取り得る最大のスコアは以下のように求まる．

$$UB_i \geq max(ps(s_1,i), ps(s_2,i), ...)$$

そのため，ニュースとツイート間の内容類似度の上限値は以下のようになる．

$$UB(s,u) = \sum_{i \in s \cap u} UB_i \geq score(s,u)$$

よって，$UB(s,u)$ よりも高い閾値をもつニュースに対しては，実際の類似スコアの計算を行う必要がない．ニュース s の閾値を θ_s とした時，実際の類似スコアの計算が必要なニュースは以下の条件を持つニュース s となる．

$$UB(s,u) \geq \theta_s \quad (1)$$

次に計算が必要なニュースを高速に検索することを考える．ポスティングリストから上記の条件 1 を満たしており，その中で最も ID が小さいニュースを高速に発見するために平衡二分木を利用する．各転置リスト毎に以下のような構造を持つ平衡二分木を作成する．

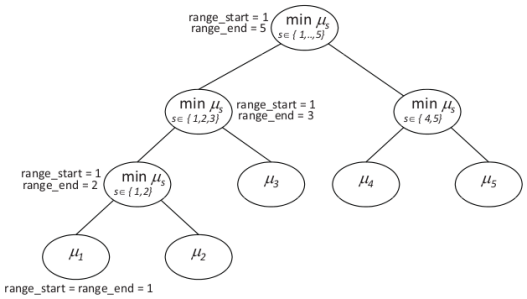


図 1: 平衡二分木 ([9] より引用)

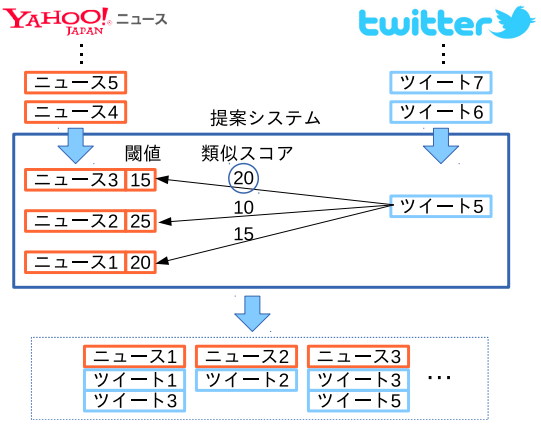


図 2: 全体の流れ

- リーフノードはポスティング内のニュース ID に対応した μ_s を持つ
- 中間ノードは子ノードの μ_s を比較し，小さいほうの μ_s を持つ
- 中間ノードは配列における始まりの位置と終わりの位置の情報を持つ

平衡二分木の例を図 1 に示す．また，平衡二分木を探索するための擬似コードを Algorithm2 に示す．

DAAT for pub-sub with skipping の処理の流れを説明する．まず，DAAT と同様に各ポスティングリストの先頭のポスティングをカレントポジションとする．カレントポジションのニュース ID を用いて各ポスティングリストを昇順にソートする．ポスティングリストの昇順に UB_i を加算していき，そのたびに平衡二分木を用いて，現在のポスティングリストから条件 1 を満たすニュース ID を探索する．条件 1 を満たすニュース ID の中で最小のニュース ID をピボットドキュメントとし，ピボットドキュメントをもつポスティングリストをピボットリストとする．その後，ピボットドキュメントの ID が，一番先頭のリストのカレントポジションのニュース ID と同じであれば，ピボットドキュメントのニュースの類似スコアを計算する．ID が異なる場合は，ピボットドキュメントの ID，もしくはそれ以上の ID まで全てのカレントディレクトリを移動する．こうすることで実際の類似スコアの計算を行うポスティング数を減らし，計算速度を向上させる．

この手法の問題点としては，転置リスト毎に平衡二分木を作成する必要があり，ニュースの追加・削除に伴う木構造の更新処理に時間がかかってしまう点が挙げられる．

5. 提案手法

本節では提案手法について説明する．提案手法ではニュースとツイートを入力として継続的に受け取り続け，ニュースとそのニュースに関連のあるツイートの結びつけを行う．提案手法の全体の流れを図 5.1 に示す．

Algorithm 2: next([9] より引用)

Input : $pos \in [1, |L_i|]$
Input : UB
Output: $next(L_i, pos, UB)$

```
1  endIndex ← findMaxInterval(I_i.root)
2  if endIndex = |L_i| then
3      return ∞
4  if endIndex =⊥ then
5      return pos
6  Procedure findMaxInterval()
7      if node.μ > UB then
8          return node.range_end
9      if isLeaf(node) then
10         return ⊥
11     p ←⊥
12     if pos ≥ node.left.range_end then
13         p ← findMaxInterval(node.left)
14         if p < node.left.range_end then
15             return p
16     q ← findMaxInterval(node.right)
17     if q ≠⊥ then
18         return p
```

まず，ニュースが配信されると，ニュースに対して類似スコアの閾値を定める．この時，ニュースが配信されるよりも過去のツイートを用いて閾値を定める．次に，ツイートが投稿されると，各ニュースとツイートとの類似スコアを計算する．類似スコアがニュースの閾値を超えていた場合，そのニュースとツイートの結びつけを行う．類似スコアの計算及び閾値との比較には，転置インデックスを使った手法を用いることによって効率的に処理を行う．

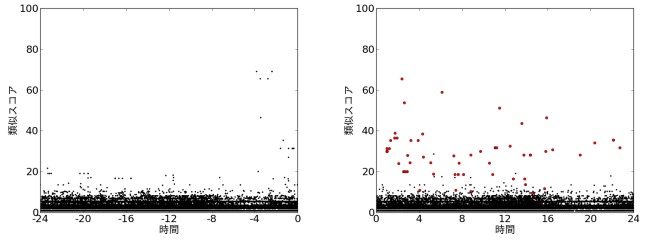
まず，5.1 節で，ニュースに対して閾値を定める方法について述べる．その後，5.2 節で，転置インデックスを用いた類似スコア計算及び比較方法について説明を行う．

5.1 閾値の設定

ここでは，ニュースに対して類似スコアの閾値を定める方法について説明する．提案手法ではニュースが配信された時点で閾値を定める．閾値の設定にはニュースの投稿時刻よりも過去に投稿されたツイートを使用する．以降，ニュースの投稿時刻よりも過去に投稿されたツイートを「事前ツイート」，後に投稿されたツイートを「事後ツイート」と呼ぶ．まず，5.1.1 節で，事前調査によってわかったニュースとツイート間の特徴について説明する．その後，5.1.2 節で，閾値の導出方法について述べる，

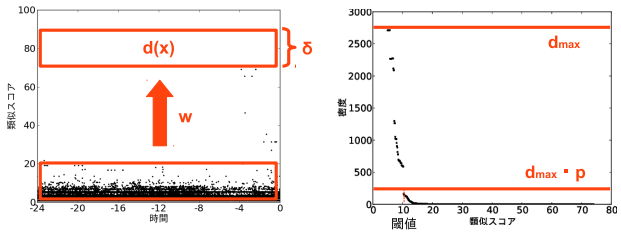
5.1.1 事前調査

閾値の定め方を決定するにあたり，各ニュースに対してツイートがどのように投稿されているか調査を行った．図 3 は，6 月 9 日に Yahoo!ニュースで配信された「<パキスタン>カラチの国際空港，武装集団が襲撃 9 人死亡」のニュースに対して，前後 1 日のツイートとの類似スコアを計算したものである．赤い点は 3.2 節



(a) 事前ツイート　(b) 事後ツイート

図 3: ツイートの分布



(a) 事前ツイートの分布　(b) ツイートの密度

図 4: 閾値の定め方

のデータセットにおける正解データであり，評価者によってニュースの内容と関連があると判断されたツイートである．黒い点はそれ以外のツイートを表している．図 3 から，ニュースが配信された時刻の前後のツイートには以下の 2 つの特徴があることがわかる．

1. ニュースと関連のないツイートは類似スコアが低く，固まって分布している
2. ニュースと関連のないツイートの分布は，ニュースの配信時刻の前後であまり変化しない

これらの特徴は他のニュースに対しても同様に見られた．提案手法では，この 2 つの特徴を利用してニュースに閾値を定める．

5.1.2 閾値の導出

閾値の導出にはツイートの密度の変化を利用する．類似スコア x に対するツイートの密度 $d(x)$ を以下のように定義する．

定義 1 (密度)

$$d(x) = \frac{|\left\{u \in U_s | (x - \frac{\delta}{2}) \leq score(s,u) < (x + \frac{\delta}{2})\right\}|}{\delta}$$

δ は範囲を表しており，密度 $d(x)$ は $x - \frac{\delta}{2}$ 以上 $x + \frac{\delta}{2}$ 未満の類似スコアを持つツイートの数を δ で割ったものである．U_s は事前ツイートの中で $score(s,u) > 0$ のツイート集合である．

5.1.1 節の特徴 (1) より，ツイートの密度を類似スコアの低い順に計算し，ツイートの密度が大きく減少した類似スコアを閾値とすることで，ニュースと関連のあるツイートのみの抽出を行う．また，ツイートの密度の計算には事前ツイートを利用する．これは 5.1.1 節の特徴 (2) より，事前ツイートにおけるツイートの密度と事後ツイートにおけるツイートの密度が大きく変化しないためである．

閾値導出のアルゴリズムを Algorithm3 に示す．事前ツイート U の中から，$score(s,u) > 0$ のツイートを抽出し U_s とする．この U_s に対して，類似スコア $x = 0$ から最大の類似スコアをもつツイートまでウィンドウ幅 δ，スライド幅 w で密度 $d(x)$ の計算を行う．この時の最大の密度を d_{max} とした時，類似スコアを昇順に調べ，以下の条件を満たす最初の x を閾値とする．

$$d(x) < d_{max} \cdot p$$

p は d_{max} からどの程度密度が減少した位置を閾値にするかを決めるためのパラメータである．

図 4 は，「＜パキスタン＞カラチの国際空港，武装集団が襲撃 9 人死亡」のニュースに対して閾値を導出する様子を表したものである．図 4(a) はニュースが配信された時刻に対して過去 24 時間のツイートを表したものであり，各点がツイートを表している．縦軸がニュースとツイートの類似スコア，横軸がニュースが配信された時間を 0 とした時の相対的な時間を表している．図 4(a) のニュースに対して密度 $d(x)$ の計算を行った結果を図 4(b) に示す．図 4(b) において，最大の $d(x)$ を d_{max} とし，$d_{max} \cdot p$ を初めて下回った類似スコアを閾値としている．

Algorithm 3: 閾値の導出

```
Input  : ニュース s, 事前ツイート U
Input  : Θ = [θ_{s_1}, θ_{s_2}, ..., θ_{s_n}]   // θ_s はニュース s の閾値
Input  : パラメータ p, ウィンドウ幅 δ, スライド幅 w
Output: 更新された Θ
U_s ← φ
for u ∈ U do
    if score(s,u) > 0 then
        Add u to U_s
U_s を score(s,u) によって昇順にソート
score_max ← max_{u∈U_s} score(s,u)
D ← φ
x ← 0
while x ≤ score_max do
    while score(s,u_i) < x + δ/2 do
        i ← i + 1
    while score(s,u_j) ≤ x − δ/2 do
        j ← j + 1
    d_x ← (i − j)/δ
    D に d_x を加える
    x ← x + w
D を x によって昇順にソート
d_max = max_{0≤x≤score_max} d_x
d_p ← d_max · p
for d_x ∈ D do
    if d_x < d_p then
        θ_s ← x
        Θ に θ_s を加える
        return Θ
```

5.2 ツイートの取得

ここでは，転置インデックスを用いて類似スコアの計算及び閾値との比較を行う手法として DAAT for variant thresholds (以降，DAAT/VAT と呼ぶ) を提案する．節の DAAT for pub-sub with skipping では，転置リストを ID 順にソートを行っていた．DAAT/VAT では各ポスティングリストのポスティングを閾値を使って昇順にソートし，DAAT を用いて処理を行う．

DAAT/VAT では，ツイートの単語を用いて上限値の計算を行い，その上限値以上の閾値をもつニュースとの類似スコアの計算

Algorithm 4: DAAT/VAT

Input : ツイート u
Input : $\Theta = [\theta_{s_1}, \theta_{s_2}, ..., \theta_{s_n}]$ // θ_s はニュース s の閾値
Input : $R_{s_1}, R_{s_2}, \cdots, R_{s_n}$
Output: 更新された $R_{s_1}, R_{s_2}, \cdots, R_{s_n}$

```
ツイート u に含まれる単語 t の転置リスト L_1, L_2, ..., L_n
UB = 0
for i ∈ [1, 2, ..., |u|] do
    UB = UB + UB_i
while true do
    s ← {L_i.cur|min_{1≤i≤|u|} θ_{L_i.cur}}
    if UB < θ_s then
        return
    score ← 0
    for i ∈ [1, 2, ..., |u|] do
        if L_i.cur = s then
            score ← score + L_i.curPs
            L_i のカレントポジションを 1 つ進める
    if score ≥ θ_s then
        R_s に u を加える
```

を省く．上限値は以下のようになる．

$$UB(u) = \sum_{i \in u} UB_i$$

DAAT for pub-sub with skipping では閾値が動的に変化することを想定していたため，ニュースの ID を用いてポスティングリストのソートを行っていた．そのため，平衡二分木を用いてポスティングリスト内を探索し，条件 1 を満たすニュースを探す必要があった．しかし，5.1.2 節の提案手法では閾値は静的であるため，閾値でポスティングリストをソートすることが可能である．そのため，各ポスティングリストを先頭から順に上限値より小さな閾値をもつニュースを探索するだけで良くなり，平衡二分木を作成する必要がなくなる．平衡二分木を探索を行わずに，単純な処理で計算を行うニュースの数を減らせるため，高速なマッチングが行える．また，ニュースが与えられたときに，平衡二分木の更新処理が発生しないため DAAT for pub-sub with skipping に比べて更新処理を高速に行うことができる．

以下に提案手法のアルゴリズムを示す．初めに，$UB(u)$ の計算を行う．各転置リストの最初のポスティングリストをカレントポジションとする．カレントポジションにおいて最も閾値が小さいニュースが，$UB(u)$ より小さな閾値である間は，DAAT を用いて類似スコアの計算を行う．カレントポジションにおいて最も閾値が小さいニュースが，$UB(u)$ より大きな閾値になった場合，処理を終了する．

6. 評価実験

本節では，提案手法の精度と性能を実験により評価する．実験環境として，Intel(R) Core(TM) i7-2600 CPU @ 3.40GHz と 32GB のメモリーを使用した．まず，6.1 節で，提案手法によって適切な閾値を定められているかを検証する．次に，6.2 節で，ニュースの追加・削除処理に伴う転置インデックスの更新処理時間を評価する．最後に，6.3 節で，ツイートの処理に要する時間の測定を行う．

6.1 精度

ここでは，提案手法によって閾値を設定した場合に，ニュースと関連のあるツイートを適切に取得できているかを検証する．データセットとして 3.2 節のデータを用いて実験を行った．各ニュースが配信されてから 1 日分のツイートを提案手法によって取得する．ニュース毎に取得したツイートに対して正解データを使って適合率と再現率を調べ，F 値を計算する．比較として，全てのニュースに対して一律の閾値を定めた場合（以降，ベースライン手法と呼ぶ）と，提案手法を用いて閾値を定めた場合において取得したツイートを比較する．ベースライン手法においては類似スコアの閾値を 20，提案手法は $p = 0.004, \delta = 10, w = 0.1$ とし，24 時間分のツイートを用いて閾値を定めた．

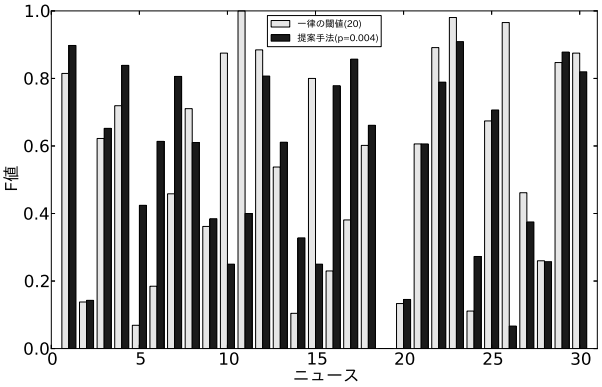


図 5: 精度比較結果

図 5 に結果を示す．提案手法では 30 件中 17 件のニュースにおいてベースライン手法よりも高い F 値を示した．ベースライン手法は，30 件中 10 件のニュースにおいて提案手法より高い F 値を示した．残りの 3 件は同じ F 値を示した．このように，提案手法の一定の有効性が確認できた．しかし，F 値以外の評価尺度の利用を含めたさらなる検討が必要である．

6.2 転置インデックスの更新

ここでは，ニュースの追加・削除処理に伴う転置インデックスの更新処理時間を調査する．まず，転置インデックスに新しいニュースを挿入する際に要する処理の時間を測定する．実験では，既存手法である DAAT for pub-sub with skipping を用いた場合と，提案手法である DAAT/VAT を用いた場合の挿入処理時間を比較する．ただし，DAAT for pub-sub with skipping と DAAT/VAT の処理の対象を同じにするため，ニュースの閾値は 5.1.2 節の手法によって定める．

データセットとして 2014 年 6 月 4 日から 2014 年 6 月 5 日に Yahoo!ニュースの速報ニュースで配信されたニュースと，2014 年 6 月 3 日 23 時 0 分から 2014 年 6 月 5 日までツイートを用いる．その中からニュースが配信された時刻が古い順に処理を行い，指定した件数分の挿入処理を行う．閾値を定める際のパラメータは $p = 0.004, \delta = 10, w = 0.1$ とし，1 時間分の事前ツイートを用いてニュースに閾値を定めた．ニュースを全て挿入し終えた後，転置インデックスからニュースの削除処理を行う．ニュースが配信された時刻が古い順に転置インデックスから削除を行い，その時に要した時間を測定する．

図 6 は，既存手法と提案手法においてニュース 1 件あたりの追加・削除に要した時間を示したものである．既存手法では，転置インデックスのポスティングリストごとに木構造を保っておく必要があり，ニュースを追加する際に木構造の更新処理を同時に行う必要がある．そのため，ニュースの件数が増加するに連れて処理時間が長くなっている．提案手法では木構造を保っておく必要がないため，既存手法と比べるとニュースの追加・削除にかかる時間が短い．

6.3 ツイートの処理

ここでは，ツイートの処理に要する時間を調べる．予め転置インデックスにニュースを追加しておき，10 万件のツイートの処理に要した時間を調べる．データセットとパラメータは 6.2 節と同様

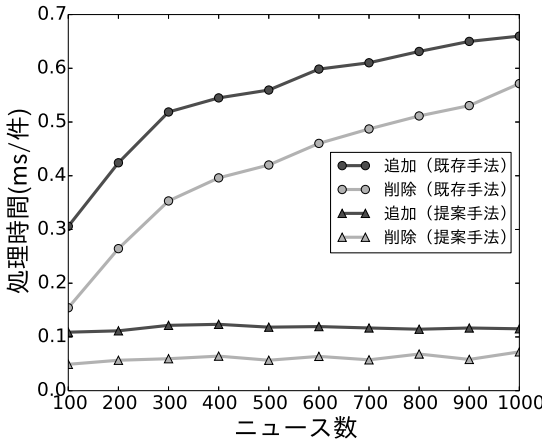


図 6: ニュースに対する処理時間

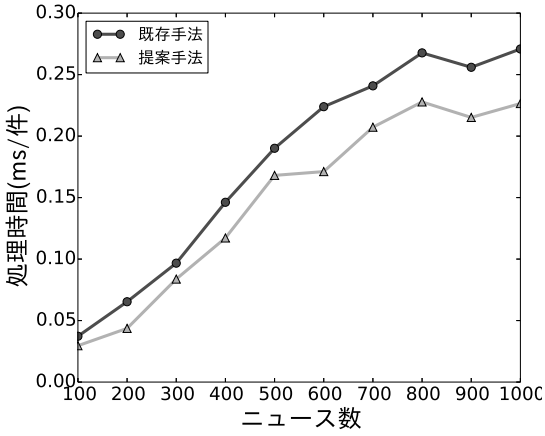


図 7: ツイートの処理時間

のものを用いる．また，先ほどと同様に DAAT for pub-sub with skipping と DAAT/VAT の処理の対象を同じにするため，ニュースの閾値は 5.1.2 節の手法によって定める．図 7 は，既存手法と提案手法においてツイート 1 件あたりに要した処理の時間を示したものである．提案手法と既存手法の処理時間はほぼ同じであるが，わずかに提案手法のほうが処理時間が短い．これは，提案手法では木構造の探索処理が発生しないためだと考えられる．

7. まとめ

本研究では，ニュースと関連のあるツイートをリアルタイムに収集するための手法を提案した．提案手法では，ツイートの密度を用いてニュース毎に閾値を定める手法と，転置インデックスを用いて閾値以上の類似スコアを持つツイートを効率的に収集する DAAT for variant thresholds を提案した．評価実験において，一律の閾値を定めた場合よりも提案手法のほうが高い精度であることを示した．また，DAAT for variant thresholds における転置インデックスの更新処理時間とツイート処理の時間を調査し，その有用性を示した．

今後の予定としては，より大規模なデータに対して評価実験を行うことが考えられる．

[謝辞]

本研究の一部は，JSPS 科研費・基盤研究（B）（26280037）および文部科学省「実社会ビックデータ利活用のためのデータ統合・解析技術の研究開発」による．

[文献]

[1] Haewoon Kwak, Changhyun Lee, Hosung Park, and Sue Moon. “What is Twitter, a Social Network or a News Media?” In WWW 2010, pp. 591-600, 2010.

[2] Alexander Pak, Patrick Paroubek. “Twitter as a Corpus for Sentiment Analysis and Opinion Mining” In LREC2010, pp.1320-1326, 2010.

[3] Mengdie Hu, Shixia Liu, Furu Wei, Yingcai Wu, John Stasko, Kwan-Liu Ma. “Breaking News on Twitter” Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, Pages 2751-2754, 2012.

[4] J. Sankaranarayanan, H. Samet, B. Teitler, M. Lieberman, and J. Sperling. “Twitterstand: news in tweets.” In GIS, pages 4251. ACM, 2009.

[5] T. Sakaki, M. Okazaki, and Y. Matsuo. “Earthquake shakes twitter users: real-time event detection by social sensors.” In WWW, pages 851860. ACM, 2010.

[6] Xuezhi Cao, Kailong Chen, Rui Long, Guoqing Zheng, Yong Yu. “News comments generation via mining microblogs” In WWW 2012, pp. 471-472, 2012.

[7] Weiwei Guo, and Hao Li, and Heng Ji, and Mona Diab. “Linking Tweets to News: A Framework to Enrich Short Text Data in Social Media” Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, pages 239249, August 4-9 2013.

[8] Qiren Qiu, Atsuo Hazeyama. “Proposal and Evaluation of a Method for Collecting Tweets Related to News Articles” IPSJ SIG Technical Report, pp.1-6, 2014-EC-31(57),

[9] A. Shraer, M. Gurevich, M. Fontoura, and V. Josifovski. “Top-k publish-subscribe for social annotation of news.” PVLDB, 6(6):385396, 2013.

[10] H. Turtle and J. Flood. “Query evaluation: Strategies and optimizations.” Information Processing and Management, 31(6):831850, 1995.

[11] A. Z. Broder, D. Carmel, M. Herscovici, A. Soffer, and J. Zien. “Efficient query evaluation using a two-level retrieval process.” In CIKM, pages 426434, 2003.

大西 誠 Sei ONISHI
2014 年筑波大学情報学群情報科学類卒業．筑波大学大学院システム情報工学研究科博士前期課程在学中．日本データベース学会学生会員．

北川 博之 Hiroyuki KITAGAWA
1978 年東京大学理学部物理学科卒業．1980 年同大学理学系研究科修士課程修了．日本電気（株）勤務の後，1988 年筑波大学電子・情報工学系講師．同助教授を経て，現在，筑波大学システム情報系教授，ならびに計算科学研究センター教授．理学博士（東京大学）．データベース，情報統合，データマイニング，情報検索等の研究に従事．日本データベース学会会長，情報処理学会フェロー，電子情報通信学会フェロー，ACM，IEEE，日本ソフトウェア科学会，各会員．