分散表現を用いた話題変化判定

芳野魁 †　　伊藤 孝行 ††
† 名古屋工業大学情報工学科　　†† 名古屋工業大学大学院情報工学専攻

1　はじめに

近年，Web 上での大規模な議論活動が活発になっているが，現在一般的に使われている”5 ちゃんねる (旧 2 ちゃんねる)” や”Twitter” といったシステムでは議論の整理や収束を行うことが困難である．Web 上での大規模合意形成を実現するために，伊藤孝行研究室は過去に大規模意見集約システム COLLAGREE[1] を開発した．COLLAGREE では，掲示板のような議論プラットフォームをベースにしており，自由に意見を投稿や返信することができる．また，他のシステムで議論の整理や収束を行うことが困難であった原因として議論の管理を行う者がいないことが挙げられるが，COLLAGREE ではファシリテーターによる適切な議論プロセスの進行を行っている．しかし，ファシリテーターは人間であり，長時間に渡って大人数での議論の動向をマネジメントし続けるのは困難である．

そこで本研究では分散表現を用いて自動的な話題変化判定を目指す．本研究の目的はファシリテーターの代わりに議論中の話題変化を判定することである．

2　関連研究

話題の遷移に基づいた文章の分割は主に人間によるテキスト全体の内容の把握の容易化や複数のテキストに対する自動分類や検索の精度向上のために研究されており，議論を専門対象とした研究は少ない．別所ら [2] は単語の共起頻度行列を特異値分解で次元を削減して作成した単語の概念ベクトルと TextTiling を用いて，トピック変化点を検出し連結された新聞記事を元の形になるように分割をする実験を行っている．しかし，TextTiling は分割する地点よりも未来の情報を使っており，リアルタイムな議論での動作には適さない．本研究は分割する地点よりも未来の情報を使うこと無く話題変化を判定する．

3　話題変化判定システム

本章では本研究で作成した話題変化判定システムの概要について説明する．

Algorithm1 を用いて話題変化判定システムの動作の流れについて説明する．本システムでは話題の変化は，発言 R と過去の発言の集合である PG の比較によって判定する．発言 R が投稿された時，PG に含まれる過去の発言と R の類似度を計算し (8 行目)，類似度が閾値を超えていた場合，同じ話題である発言集合 SG に比較した 2 発言の話題が同じであるとして登録する (9 ∼10 行目)．全ての比較が終了した後，発言 R に返信先がなく，SG が空集合である，すなわち発言 R と同じ話題である発言がない場合に話題を変化させる発言であると判定して通知を行う．

過去の発言と R の類似度は，発言内容間の類似度に発言間の時間差と返信関係を組み合わせることで総合類似度を求める．時間差評価値は発言間の時間差を議論の制限時間で割ることで最大値が 1，最小値が 0 となるように正規化したものを用いる．総合類似度に時間差評価値を導入して時間的に近いものほど総合類似度を上昇させることで，議論が基本的に少し前に発言に関連して進行されることが多いという点を考慮した．具体的には式 1 のように計算される．

Algorithm 1 話題変化判定システムの流れ

```
1:  Input : 発言 R
2:  Output : 通知判定 Notify
3:  PG = 過去の発言の集合;
4:  procedure TOPICCHANGE(R)
5:      SG = {};
6:      update(R)
7:      for Each pastR ∈ PG do
8:          sim = similarity(R,pastR)
9:          if sim > threshold then
10:             SG.append(pastR)
11:     Notify = False
12:     if SG == {} then
13:         Notify = True
14:     return Notify
```

$$total = time * tWeight + sim * (1 - tWeight) \quad (1)$$

time，sim はそれぞれ時間差評価値と発言内容の類似度を表し，tWeight は時間差評価値の重みを示しており 0 から 1 の値を取る．また，2 発言が返信関係にあった場合は時間差評価値を無視し，発言内容の類似度に補正値を加えたものを総合類似度とした．

4　発言内容の類似度計算

本章では発言内容，すなわち文字列の意味的類似度を計算する手法を説明する．

分散表現による類似度計算で精度を上昇させるためには発言内容から余分な単語を取り除き重要な単語を抽出する，または極めて短く要約することが重要である．提案手法では形態素解析エンジン MeCab と重み付けアルゴリズム okapiBM25 と LexRank を用いて発言の文章から重要単語を抽出し，抽出した単語の類似度を分散表現によって計算する．本研究では MeCab による形態素解析の結果，次の条件を満たす単語を除外している．

1. 品詞細分類に「数」を含む
2. 「読み」,「発音」が不明である
3. 品詞が「助詞」,「助動詞」,「記号」,「連体詞」のどれかである．
4. 1 文字のひらがなである
5. 品詞細分類に「接尾」または「非自立」を含む

上記の条件を満たす単語を除外することで分散表現を用いた類似度計算の精度を上昇させる．

提案手法では okapiBM25 と LexRank の 2 種類の重み付け手法を統合して発言の内容の文字列 $remark$ 中の単語に対して重み付けを行う．アルゴリズムを **Algorithm2** に示す．本研究では計算された単語重みの値が大きいものの上位 n 個までの単語を発言文章 $remark$

A Topic Change Judgment Method based on Distributed Representation
Kai Yoshino　Takayuki Ito
†Department of Computer Science, Nagoya Institute of Technology
††School of Techno-Business Administration, Graduate School of Engineering Nagoya Institute of Technology

Algorithm 2 統合重みの計算アルゴリズム

```
Input : remark 発言内容の文字列
Output : combinedWeight remark 中の単語と重みを対応付けた連想配列
Array sentList; ▷ 以前に重み付けを行った最大 n 個前までの文章のリスト
procedure calcCombinedWeight(reamrk)
    bm25Weight = calcBM25Weight(remark)▷ 単語と重みの連想配列
    for Each sent ∈ remark do ▷ remark を句点，改行コードで分割する
        sentList.append(sent)
    lexWeight = calcLexRank(sentList)
    for Each word ∈ bm25Weight. keys() do
        wordWeight = bm25Weight[word]
        if word is 固有名詞 then
            wordWeight *=4
        sentWeight = 0
        for Each sent ∈ remark do
            if word in sent then
                sentWeight+=lexWeight[sent]
        combinedWeight[word] = wordWeight*sentWeight
    return combinedWeight
```

において重要度の高い単語であるとして抽出する．それぞれの発言から抽出された単語を分散表現を用いて単語をベクトルに変換し，平均ベクトルの Cosine 類似度を取ることで発言文章間の類似度としている．本研究では分散表現として fastText[3] を用いる．

5 評価実験

5.1 実験設定

評価実験では COLLAGREE 上で行われた議論時間９０分の 2~3 名による複数の議論データを用意し，提案手法と比較手法で実験を行う．結果として提案手法のほうが分散表現を用いていることで良い精度を出せるか確認する．議論データに対し，次に述べる基準で学生にアノテーションを行ってもらった．アノテーション担当者が基準を満たすと判断した発言に”1”のタグを，満たすと思われない発言に”0”のタグを付ける．

1. それまで話題となっていた対象や事態とは異なる，新しい対象や事態への言及する発言
2. 既に言及された対象や事態の異なる側面への言及する発言
3. 議論のフェーズを移行させる (可能性の高い) 発言
4. ファシリテーターによる議論をコントロールするような発言

以上の基準に沿ってタグを付けてもらい，”1”のタグが過半数より多く付けられた発言を正解値=1，他を正解値=0 とした．

比較手法として常に予測値=1 とする手法 (比較手法 1) と発言文章を TF-IDF によって求められた単語の重みを用いてベクトル化する手法 (比較手法 2) と発言文章を LDA を用いてトピックベクトル化する手法 (比較手法 3) の 3 つを用いた．

提案手法 1 ではパラメーターは次の通りに設定した．前処理にて用いる okapiBM25 のパラメーターは $k_1 = 2, b = 0.75$ とし，LexRank では 50 個前までの文を用いた．また，重み付けを用いて文章から抽出する単語の数は 5 個とした．fastText は次元数を 100 次元とし，学習データには wikipedia ダンプデータを用いた．提案手法 2 では単語抽出を行わずに文章中の全単語の平均ベクトルで内積を取る．式 1 の $tWeight$ は 0.5 とし，総合類似度の閾値は 0.8 とした．

評価指標として適合率 (Precision)，再現率 (Recall)，F 値 (F-measure) の 3 種類の指標を用いる．

5.2 実験結果

実験結果を表 1 に示す． 実験結果より，分散表現を

手法	平均評価指標		
	Precision	Recall	F-measure
比較手法 1 (常に予測値=1)	0.257	1	0.404
比較手法 2 (TF-IDF ベクトル)	0.271	0.865	0.407
比較手法 3 (LDA ベクトル)	0.309	0.366	0.332
提案手法 1 (単語抽出あり)	0.331	0.757	0.455
提案手法 2 (単語抽出なし)	0.750	0.081	0.145

表 1: 実験結果

用いた提案手法は比較手法よりも平均的に高い F 値を示している．また，単語抽出を行うことで再現率と F 値が大きく上がることが確認できた．

6 まとめ

本研究ではファシリテーターの負担を軽減するために新たに投稿された発言を調査し，分散表現を用いて自動的に話題の変化を判定する手法を提案した．COLLAGREE 上で行われた議論のデータを対象にした提案手法の評価実験の結果，提案手法が比較手法よりも高い精度で話題変化を判定できることが確認できた．しかし，本研究では実際の議論においてのファシリテーターによる評価が行われていない．また，提案手法によって求められる発言文章間の類似度は抽出される単語に大きく依存し，常に適切な単語抽出を行うことは困難である．

したがって，今後の課題として COLLAGREE での実装及び実証実験を行うことと単語抽出への依存を改善する，または単語抽出の精度を上昇させることが挙げられる．

参考文献

[1] 伊藤孝行, et al. ”多人数ワークショップのための意見集約支援システム collagree の試作と評価実験”. 日本経営工学会論文誌, pp. 83–108, 2015.

[2] 別所克人ほか. 単語の概念ベクトルを用いたテキストセグメンテーション. 情報処理学会論文誌, Vol. 42, No. 11, pp. 2650–2662, 2001.

[3] Bojanowski and Piotr et al. ”enriching word vectors with subword information”. *arXiv preprint arXiv:1607.04606*, 2016.