The 32nd Annual Conference of the Japanese Society for Artificial Intelligence, 2018

ファシリテーター支援を目的としての分散表現を用いた話題変化判定

Topic change judgment using distributed representation for facilitator assistance

芳野 魁[*1]
Kai Yoshino

伊藤 孝行[*1]
Takayuki Ito

[*1] 名古屋工業大学大学院 情報工学専攻
Nagoya Institute of Technology, Department of Computer Science

In order to reduce the burden on the facilitator in the large-scale opinion concentration system COLLAGREE, we propose a topic change judgment method using distributed representation. The proposed method extracts important words from utterances using extractive summaries and calculates similarities between utterances using distributed expressions. Experimental results show that the proposed method has higher performance than the comparative method and word extraction contributes to performance improvement.

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. 一般的には"5 ちゃんねる(旧 2 ちゃんねる)" や"Twitter" といったシステムが使われている. しかし, 既存の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 を開発した[伊藤 et al., 2015]. COLLAGREE は自由に意見を投稿や返信することができる掲示板のような議論プラットフォームをベースにしている. 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. 前述の問題に対応するため, 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, 長時間に渡って大人数での議論における話題動向をマネジメントし続けるのは困難である.

そして, 分散表現は自然言語処理の分野において多くの研究で使われている. 分散表現は単語の共起頻度に基いて類似度を計算しているため, 同じ文脈で出現することの多い対義語に対応できない点が弱点とされる. しかし, 議論中の話題においては対義語かどうかは関係なく同じテーマ(話題)に沿っているかが重要である. 分散表現における類似度計算は議論における話題の繋がりと似ているため, 分散表現の対義語にできないというデメリットを無視して話題変化判定ができると考えられる.

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

2.1 COLLAGREE

(1) 概要

COLLAGREE は各ユーザーが自由なタイミングで意見を投稿や返信することのできる掲示板のような議論プラットフォームをベースにしている. 図 1 に COLLAGREE での実際の議論画面を示す. COLLAGREE のような議論掲示板は基本的には1つの議論テーマに対して関連するテーマを扱った複数のスレッドから構成される. スレッドとはある特定の話題・論点に関する1つのまとまりを指す.

図 1:COLLAGREE の議論画面

(2) ファシリテーター

COLLAGREE ではファシリテーターと呼ばれる人物が議論のマネジメントを行っている.

ファシリテーターは議論そのものには参加せずにあくまで中立的な立場から活動の支援を行うため, 自分の意見を述べたり自ら意思決定をすることはない.

連絡先:芳野魁 yoshino.kai@itolab.nitech.ac.jp

ファシリテーターの基本的な役割として"議論の内容の整理", "議論の脱線防止", "意見の促し"等が挙げられる.

ファシリテーターが様々な論点に対する発言を促すことによって議論が発散するため, 十分な議論が行われる.

ファシリテーターの存在や手腕が合意形成に強く影響を与えることや Web 上での議論においてファシリテーターが有用であることが示されている[田中 et al., 2015][伊美 et al., 2015].

2.2 話題遷移検出

話題の遷移に基づいた文章の分割が主に人間によるテキスト全体の内容把握を容易にすることや複数のテキストに対する自動分類や検索精度向上を主な目的として研究されている.

別所ら[別所 et al., 2001]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[Hearst, 1997]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.

本研究と別所らの研究は両研究とも文字列間の類似度を話題に基いて計算している点で関連している. しかし, TextTiling はグラフ全体の中からブロック(複数の単語を連結したもの)間類似度の小さい地点から分割を行うため, 分割する地点よりも未来の情報を使っている. 故に, 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

3.1 システム動作の流れ

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.

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:  P G = 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
```

最初に, 新しく投稿された発言 R で使われている単語の情報を 4 章で説明する重み付けで用いるために単語の出現回数等の更新を行う(6 行目). そして, 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 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, 話題を変化させる発言であると判定する.

3.2 話題変化判定

発言間の類似度計算は次の3段階で行われる.

(1) 前処理

発言内容の類似度計算の精度を上昇させるために, 本研究では前処理としてストップワードの除去や単語の重み付けを行う. 前処理の詳細については 4 章で述べる.

(2) 発言内容の類似度計算

前処理の情報や分散表現を用いて発言内容文の類似度計算を行う. 文章間の類似度計算の手法については 4 章で詳しく述べる.

(3) 総合類似度計算

計算された発言の文章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. 時間差評価値は式(1)に示すように, 発言間の投稿時間差を最大時間差で割り 1 から引くことで時間差評価値 $tValue$ は 0 から 1 の範囲となる. 2 発言間の時間差が小さいほど関連が強いとみなされて 1 に近くなる.

$$tValue = 1 - \frac{new.created - old.created}{maxTime} \quad (1)$$

$maxTime$は最大時間差を表す数値である. $maxTime$は基本的に議論の制限時間を用いる. $x.created$は発言xが投稿された時間を表す. 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(2)のように計算される.

$$total = tValue * tWeight + sim * (1 - tWeight) \quad (2)$$

$time$, simはそれぞれ時間差評価値と発言内容の類似度を表す. $tWeight$は時間差評価値の重みを示す. $tWeight$は 0 から 1 の値を取る. また, 2 発言が同じスレッドに属していた場合は何らかの関連があると考えられる. 故に, 2 発言が同じスレッドに属してい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. 話題変化判定システム

4.1 ストップワード除去

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か極めて短く要約することが重要である. 本研究では形態素解析エンジン 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た.

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

4.2 重み付け

提案手法ではokapiBM25[Robertson, 1995]とLexRank[Erkan, 2004]の2種類の重み付け手法を統合して発言文章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.

Algorithm 2 統合重みの計算アルゴリズム

```
1:  Input : remark
2:  Output : combinedWeight
3:  Array sentList;
4:  procedure calcCombinedWeight(reamrk):
5:    bm25Weight = calcBM25Weight(remark)
6:    for Each sent ∈ remark do
7:      sentList.append(sent)
8:    lexWeight = calcLexRank(sentList)
9:    for Each word ∈ bm25Weight.keys() do
10:     wordWeight = bm25Weight[word]
11:     if word is 固有名詞 then
12:       wordWeight *=4
13:     sentWeight = 0
14:     for Each sent ∈ remark do
15:       if word in sent then
16:         sentWeight+=lexWeight[sent]
17:     combinedWeight[word] = Weight*sentWeight
18:   return combinedWeight
```

$remark$は発言内容の文字列を意味し, $combinedWeight$は $remark$ 中の単語と重みを対応付けた連想配列を表す. $sentList$ には重み付けを行った最大 n 個前までの文章が入る. $remark$を句点, 改行コードで$sent$に分割する(6 行目). 本研究において固有名詞は文章の中で重要な役割を果たす可能性が大きいと考えたため, 固有名詞の単語重みを 4 倍にしている. (12行目). wordを含む全文章の重みの合計を求める(14~16行目). そして, okapiBM25 による単語重みを掛け合わせたものをwordの統合重みとしている(17 行目).

単語重みに単語を含む文章の重みを掛け合わせることで感動詞のような単語を含む文章そのものは重要でないが, 重要性が高いと判断されてしまう単語が選ばれる可能性を下げている.

4.3 重み付け

本研究では計算された単語重みの上位 n 個までの単語を発言文章$remark$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た後, 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[Bojanowski et al., 2016]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.

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様な発言

過半数のアノテーション担当者が基準を満たすと判断した発言を話題変化発言とした. また, 比較手法として以下に述べる3つを用いた.

- [比較手法 1]:常に話題変化発言として判定する
- [比較手法 2] : 発言文章を TF-IDF による単語の重みを用いてベクトル化したものを fastText の代わりに用いる
- [比較手法 3] : 発言文章を LDA を用いてトピックベクトル化したものを fastText の代わりに用いる

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(2)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

実験結果として各手法の F 値を表 1 に示す. 各手法の再現率と適合率及び 2 つの値の差を表 2 に示す.

手法	F-measure
比較手法 1 (常に予測値=1)	0.404
比較手法 2 (TF-IDF ベクトル)	0.487
比較手法 3 (LDA ベクトル)	0.385
提案手法 1 (単語抽出あり)	0.515
提案手法 2 (単語抽出なし)	0.086

表 1:実験結果 1

手法	平均評価指標		
	Precision	Recall	Difference
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)	0.489	0.558	0.069
比較手法 3 (LDA ベクトル)	0.573	0.317	0.256
提案手法 1 (単語抽出あり)	0.515	0.552	0.037
提案手法 2 (単語抽出なし)	0.833	0.046	0.787

表 2:実験結果 2

表 1 が示すように, 提案手法 1 は他のどの手法よりも高い F 値を出している. 一方で, 表 2 が示すように, 提案手法では再現率と適合率の間の差が 0.037 で最も小さくなっている. 以上から, 適合率と再現率のバランスが最も良かったことが他の手法よりも高い F 値に繋がったと思われる.

また, 表 1 が示すように, 提案手法 1 は提案手法 2 よりも高い F 値を出している. 一方で, 表 2 が示すように提案手法 2 は非常に高い適合率を示すと同時に, 非常に低い再現率を示している. 提案手法 1 と提案手法 2 の間の違いは単語抽出を行うかどうかだけなので結果に影響を与えたのは明らかに単語抽出である. 単語抽出が結果に影響を与えた理由として, 単語抽出によって平均を取る単語ベクトルの数が減ったことが考えられる. すなわち, 平均を取る単語ベクトルの数が多くなることで単語に関係なく結果的に平均ベクトルが全て似たようなものとなったため, 発言の平均ベクトルの差が小さくなってしまったと想定できる.

6. おわりに

本研究ではファシリテーターの負担を軽減するために, 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, 現在の状態では提案手法の適合率は高いとは言えない. 原因としては常に適切な単語抽出を行うのは困難であることが考えられる.

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が挙げられる. また, 単語抽出の精度を上昇させること, もしくは, Word Mover's Distance[Kusner et al., 2015] の様な単語抽出を行わない手法への変更を行うことが挙げられる.

参考文献

[伊藤 et al., 2015] 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, pp83-108, 2015.

[田中 et al., 2015] 田中恵, 伊藤孝紀, 伊藤孝行, 秀島栄三. ”ファシリテー タに着目した合意形成支援システムの検証と評価”. デザイン学研究, pp. 67–76, 2015.

[伊美 et al., 2015] 伊美裕麻, 伊藤孝行, 伊藤孝紀, 秀島栄三. “オンラインファシリテーション支援機構に基づく大規模意見集約システム collagree ―名古屋市次期総合計画のための市民議論に向けた社会実装”, 情報処理学会論文誌, Vol. 56, No. 10, pp. 1996–2010, 2015.

[別所 et al., 2001] 別所克人, et al. “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.

[Hearst, 1997] Marti A Hearst. “Texttiling: Segmenting text into multi-paragraph subtopic passages.” Computational linguistics, Vol. 23, No. 1, pp. 33–64, 1997.

[Robertson, 1995] Stephen E Robertson, Steve Walker, Susan Jones, Micheline M Hancock-Beaulieu, Mike Gatford, et al. Okapi at trec-3. Nist Special Publication Sp, Vol. 109, p. 109, 1995.

[Erkan, 2004] Gunes Erkan and Dragomir R. Radev. ”lexrank: Graph-based lexical centrality as salience in text sum- marization”. Journal of Artificial Intel ligence Re- search, pp. 457–479, 2004.

[Bojanowski et al., 2016] Bojanowski and Piotr et al. ”enriching word vec- tors with subword information”. arXiv preprint arXiv:1607.04606, 2016.

[Kusner et al., 2015] Matt Kusner, Yu Sun, Nicholas Kolkin, and Kilian Weinberger. From word embeddings to document distances. In International Conference on Machine Learning, pp. 957–966, 2015.