E-013

時系列推移を考慮した鉄道トラブルに関わる tweet 抽出手法の検討

Tweets Extraction Method about Train Trouble with Time-series Consideration

鳥海 心† 宮﨑 太郎‡ 後藤 淳‡ 山田 一郎‡ 八木 伸行†
Shin Toriumi Taro Miyazaki Jun Goto Ichiro Yamada Nobuyuki Yagi

1. はじめに

現在、放送局では tweet などのソーシャルメディアから有用な情報を収集する取り組みを進めている[1]。事件・事故の第一報を得るために有効であるが、手作業による部分が多く、大きな労力がかかっている。我々はこれらを自動化するため、鉄道トラブルに関する tweet の自動抽出手法について報告した[2]。提案した tweet 自動抽出手法では、鉄道トラブルが発生した際、その状況に合わせて自動でクエリ拡張し、多くの tweet を収集しランキングした。

本稿では、提案手法[2]について、トラブル発生が検知された時刻からどれくらい早い段階で必要な情報が自動で抽出できるかを評価したので報告する。

2. 提案手法

提案手法では、図 1 のように①対象トラブルに合わせたクエリ拡張、②分類器を用いた tweet のランク付け、の 2 段階で処理し、トラブルを表している有用な tweet を獲得する。以下で各処理の詳細を説明する。

2.1 対象トラブルに合わせたクエリ拡張

対象トラブルごとに設定した期間内の全 tweet から、路線名をクエリとしたキーワードマッチングで tweet を抽出する。抽出した全ての tweet を形態素解析し、tweet 内の助詞、助動詞を除く全単語分の TFIDF を計算する。その後、抽出した全 tweet に出現する単語の TFIDF を足し合わせる事により、トラブルを表す特徴的な単語をランキングする。ランキングした単語の上位 15 位に含まれる名詞を抽出し、ストップワードを削除したものを拡張したクエリとする。

2.2 分類器を用いた tweet のランク付け

拡張したクエリを含む tweet を抽出し、それらを分類器を用いてランキングし、鉄道トラブルを表す有用な tweet を獲得する。

まず、拡張したクエリのいずれかを含む tweet をキーワードマッチングにより抽出する。次に、分類器により、それぞれの tweet にスコアを付け、スコアに基づいて抽出した tweet をランキングする。分類器の学習には、放送局で収集している鉄道トラブルに関する tweet を正例、ランダムサンプリングした tweet を負例とする。分類器には SVM(Support Vector Machine)と NN(Neural Network)の 2 つを用意し、評価実験で比較をする。SVM は分離平面からの距離、NN では出力の重みで tweet をランキングする。分類器に入力する素性には、tweet に含まれる単語の分散表現の加算平均を用いた。

† 東京都市大学 Tokyo City University
‡ NHK 放送技術研究所
NHK Science & Technology Research Laboratories

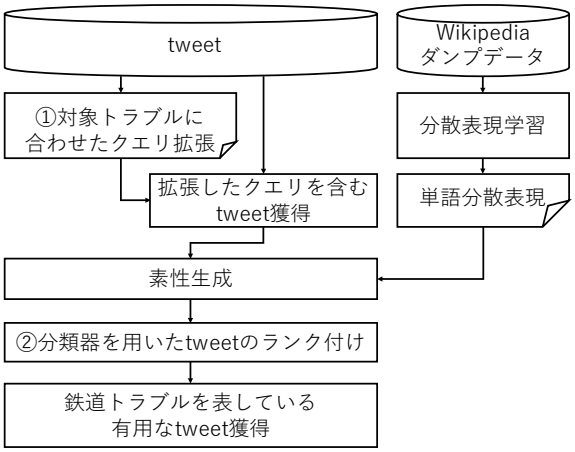

図 1 提案手法の概要

3. 評価実験

提案手法の効果を確認するため、評価実験を行った。提案手法により出力された tweet を 1 名の被験者により評価した。いかに少ない tweet でニュースに必要な情報を獲得できるかを評価するため、出力した tweet を 1 位から順に確認し、路線名、発生場所、鉄道状況、トラブル原因の 4 つの情報をそれぞれ何番目の tweet で獲得する事ができるかを調べた。

3.1 実験条件

今回、2016 年に発生した 3 つの鉄道トラブルを実験の対象とした。表 1 に鉄道トラブルの概要を示す。

クエリ拡張に用いる形態素解析には MeCab[3]を使用した。単語をランキングするための IDF の計算には、Wikipedia の 2016 年 9 月のダンプデータを用いた。

Tweet をランキングするために使用した SVM には SVM-Light[4]を使用し、多項式カーネルにより判定した。NN の実装は Chainer[5]を使用し、入力層、中間層、出力層の 3 層でネットワークを構築した。入力層は 200 次元、中間層は 100 次元、出力層は 2 次元に設定した。単語の分散表現は Word2Vec[6]を用い、IDF の計算と同様、Wikipedia のダンプデータから計算した。

解析対象とする tweet は、日本語の全 tweet のうちの 10%サンプリングを利用し、トラブル発生を検知した時刻から 1 分後、3 分後、5 分後に区切り、時系列を考慮して評価した。報道現場での第一報を捉えることを目的としているため、トラブル検知時刻から 5 分以内を期間とする。トラブルを検知した時刻は、トラブル発生後、路線名を含む tweet が 3 つ出現した時刻とする。

ベースライン手法には、tweet に出現し、かつ拡張したクエリに含まれる単語について TFIDF の値を足し合わせ、値の降順にランキングしたものを用いる。

3.2 実験結果

評価実験の結果を表 2 に示す。表内の数値は各情報を獲得できた tweet の順位を示す。“-” は該当する tweet が存在しない場合である。なお、手法の欄の BL は、ベースライン手法である TFIDF を用いたランキングを表す。

表 2 からは、1 件のトラブルにおいてトラブル検知から 1 分で状況を把握できるだけの情報を抽出できた事がわかる。残りの 2 件のトラブルでも 3 分後には情報抽出が可能であった。以上の事から、提案手法が迅速な報道のための情報抽出に有効である事がわかった。

3.3 考察

小田急線の例では、トラブル検知から 1 分ではトラブルの原因についての情報が抽出できなかったが、3 分の時点で抽出できた。本手法は、時間の経過により情報が追加された場合にも抽出できる点で、リアルタイムな情報抽出に有効な手法と言える。

提案手法として用意した NN と SVM では、性能に大きな差が見られなかった。これは、共に word2vec を用いた単語の分散表現を特徴量とした分類手法であり、この特徴量から判定できる限界によるものと考えられる。今後、性能を向上するために tweet 内での単語の出現順序も考慮したモデルを用いるなど、複雑なモデルを用いる事で、この 2 つの手法の性能差が現れると考えられる。一方、TFIDF での抽出では、拡張したクエリが含まれるものを上位に表示するため、当該の路線と無関係のものも多く上位に含まれてしまい、NN や SVM と比較し、性能が悪かった。

中央線のトラブルでは「三鷹で下り人身事故。東京〜高尾全線見合わせ。」という、路線名を含まずにトラブルについて詳細に言及している有用な tweet が上位にランクされている。これは、クエリ拡張をする事によって獲得できた tweet である。対象 tweet が少ないトラブル発生直後では、候補となる tweet を増やす事が可能となるため、リアルタイムにトラブルに関係する tweet を抽出する際に、クエリ拡張が特に初期段階では有効であると考えられる。

4. おわりに

本稿では、先行報告[2]で提案した鉄道トラブルに関する tweet の自動抽出手法の時系列抽出性能を評価した。3 件の鉄道トラブルの発生が検知された時刻から 5 分間を対象に、自動で情報を抽出した場合の性能について評価した。

結果として、2 件のトラブルにおいて検知後 1 分で状況把握が可能な情報を抽出する事ができ、残りの 1 件のトラブルに関しても 3 分後には情報抽出が可能であった。以上より、提案手法が事故検知から早い段階で多くの情報を抽出する手法として有効である事がわかった。

今後、性能の向上を図るために、同じような内容の tweet を 1 つにまとめ、グループとしてランキングする手法や、単語の出現順序を考慮する事ができる Recurrent Neural Network を用いた手法を検討する。

表 1 実験に用いた鉄道トラブル概要

日時	路線	発生場所	状況	原因
7/14	小田急線	全線	運転見合わせ	線路冠水
8/17	中央線	三鷹駅	運転見合わせ	人身事故
9/2	田園都市線	用賀駅	運転見合わせ	ガラス破損

表 2 実験結果

7/14 小田急線	手法	路線	場所	状況	原因
トラブル検知から 1 分後	BL	1	3	1	-
	SVM	1	1	1	-
	NN	1	2	1	-
3 分後	BL	2	3	2	114
	SVM	1	1	1	10
	NN	1	5	1	12
5 分後	BL	2	4	2	141
	SVM	1	2	1	13
	NN	1	3	1	25
8/17 中央線	**手法**	**路線**	**場所**	**状況**	**原因**
トラブル検知から 1 分後	BL	1	1	1	1
	SVM	1	2	1	2
	NN	2	1	1	1
3 分後	BL	1	1	3	1
	SVM	2	1	1	1
	NN	2	1	1	1
5 分後	BL	1	1	2	1
	SVM	2	1	1	1
	NN	2	1	1	1
9/2 田園都市線	**手法**	**路線**	**場所**	**状況**	**原因**
トラブル検知から 1 分後	BL	1	2	-	3
	SVM	1	1	-	3
	NN	1	2	-	3
3 分後	BL	1	1	2	2
	SVM	1	1	2	2
	NN	1	2	4	3
5 分後	BL	3	3	1	3
	SVM	2	5	1	2
	NN	2	6	1	3

参考文献

[1] 足立義則, “震災ビッグデータからソーシャルリスニングへ,” 放送メディア研究, No.11, pp.290-293, (2014).

[2] 鳥海ほか, “鉄道トラブルに関する tweet の自動抽出手法.” 第 23 回言語処理学会年次大会発表論文集, D4-1, pp.418-421, (2017).

[3] Taku Kudo, et al., “Applying Conditional Random Fields to Japanese Morphological Analysis,” in Proceedings of EMNLP 2004, pp. 230–237, (2004).

[4] Thorsten Joachims, “Making Large-scale SVM Learning Practical. Advances in Kernel Methods - Support Vector Learning,” MIT-Press, (1999).

[5] Seiya Tokui et al., “Chainer: a Next-Generation Open Source Framework for Deep Learning,” in Proceedings of NIPS 2015 workshop, (2015).

[6] Tomas Mikolov, et al., “Efficient estimation of word representations in vector space,” arXiv preprint arXiv:1301.3781, (2013).