トラブルを見つける

De Saeger Stijn　　鳥澤 健太郎
情報通信研究機構　(stijn@nict.go.jp)
北陸先端科学技術大学院大学 (torisawa@jaist.ac.jp)

概要 本研究では、モノとその利用に関する障害やトラブルを表す語彙のペアを獲得する方法を提案する。モノとその利用に関する障害やトラブルと呼んでいるものの具体例としては「ディスニーランド・身長制限」、「Wii・ストラップ問題」があげられる。本研究で提案する方法は二段階からなる。第一段階では教師あり学習を用い、特定の構文パターンに基づきトラブルを表す言語表現を大規模な Web コーパスから抽出する。第二段階では抽出したトラブル表現を動詞の共起パターン等により様々なモノと関連付ける。実験を通して提案手法の有効性を示す。

1　はじめに

あるモノや人工物ついて Web 上の検索エンジンで調べる時に、そのモノの利用に関しての障害・危険・トラブル等という、ユーザーにとってきわめて重要な情報の一種がある。しかし、そういった障害・危険・トラブルは基本的にモノの利用に際し「想定外」の状況で起こるので、ユーザーが簡単には検索できない情報である。例えば、ディスニーランドへの旅行を計画しているユーザーが Web 上で「東京ディスニーランド」について調べる時に、年齢や身長等というアトラクションの様々な利用制限について教えられると有用であろうが、現在の検索エンジンではそれらのトラブル表現を具体的にキーワードとして指定しない限りそういった情報が得られない。一方、企業も定期的な調査により、製品の不良や客の様々な不満を早期に発見することにより、大規模なリコールの高いコストや機会損失を避けることもできるかもしれない。

本研究ではモノとその正常な利用や楽しみ方の妨げになるトラブルを “object-trouble pair” と呼び、検索エンジンのフロントエンド等に役立つ知識資源の構築に向けて object-trouble pair の自動獲得方法を提案する。この方法は二段階からなる。まず大規模な Web コーパスから様々な構文パターンを手がかりとした教師あり学習の手法により一般的にトラブルになりやすい表現を獲得する。それからこれらのトラブル表現を様々なオブジェクトと組み合わせ、ペアの共起出現度等を考慮しながらトラブル表現がそのオブジェクトの用途に関するトラブルになるかどうかを推定する。

2　背景

本研究で目標としているのは「関係抽出」(*relation extraction*) という、非構造化テキストデータから関係知識を自動的に獲得するタスクである。抽出の対象になる意味的関係は、例えば “person-affiliation” と “organization-location” (Zelenko ら [8]; Agichtein ら [1])、“author-title” (Brin, [3])、“reaction”, “production” (Pantel ら [7]) や “textual entailment” (Dagan ら [4]) 等が例としてあげられる。日本語の場合では、[6] で乾らが “part-of”, “effect”, “condition-for”, “cause” 等という様々な事態関係の自動獲得について発表した。

これらの研究との主な違いは、本研究で狙っているトラブル関係が抽象的で、文の表層構造には決定的な手がかりが比較的に少ない。

3　トラブル表現の手がかり

第一段階では一般のトラブル表現を大規模な Web コーパスから抽出する。本研究でトラブル表現の手がかりとして取り上げる要素について以下に説明する。

LSPH トラブル表現の自動獲得は下位語の獲得の特別な場合と見なせる。獲得したい表現が一般的に「トラブル」、「障害」、「災難」等の下位語であると考えると、文中の「X のようなトラブル」、「X 以外の障害」等という、下位語獲得のための典型的なパターン (Hearst [5], Ando ら [2]) はトラブル表現の手がかりと考えることができる。以降、こうした手がかりを「LSPH」 (“lexico-syntactic patterns for hyponymy”) と省略する。

しかし、これらの構文パターンこれだけではトラブル表現の決定的な手がかりにならないケースが多い。なぜ

かというと、例えば「バッテリーのようなトラブル」や「頭以外の障害」という、漠然とした表現でトラブルに言及するパターンが多いからである。

DNV・DAV 本研究でトラブルは一般的にモノの正常な用途の妨げになる要素と定義する。例えば、「病気」や「渋滞」は人々の日常生活や行き先の妨げになる。この定義に基づいてトラブル表現とその係先になる動詞との様々な共起パターンをもう一種の手がかりとして考慮する。具体的にはトラブル表現の肯定的手がかりと否定的手がかりになる動詞との係り受け関係を区別する。例えば、トラブル表現はよく以下のような文に出現するであろう。

- "X で Y できなかった"
- "X で Y に行かなかった"

ここから X が Y の妨げになるトラブルであると推測できる。このパターンの共通点は (名詞 X+ 助詞:で) というトラブル表現 X と否定動詞 Y の係り受け関係であり、本研究ではこうしたパターンを「DNV」("dependencies with negative verbs") と呼ぶ。

しかし、「で」という助詞は「理由」の他にも「手段」、「場所」等を表す機能を持つため、トラブル表現の手がかりとしてはこれだけでは精度が低い。そのため、トラブル表現の否定的手がかりとして以下の係り受けパターンも考慮する。

$$\text{(名詞 } X\text{+ 助詞:で)} \rightarrow \text{肯定動詞} \qquad (1)$$

こうしたパターンを DAV("dependencies with affirmative verbs") と呼ぶ。もしある DNV パターンに現れる表現の X が実際にトラブル表現でなければ、その X を含む DAV パターンもある程度同じ頻度でデータに現れるはずという仮定である。例えば、「車で街に行かなかった」という文には、「車」がトラブル表現でないので「車で街に行った」という文もほぼ同頻度で見つかるはずである。つまり、DAV は DNV パターンで獲得する表現のフィルターとして機能する否定的手がかりとも言える。教師あり学習に際しこうした DAV の否定的手がかりによりトラブル表現の認識精度が大幅に向上すると考えられる。

4 獲得方法

本研究の獲得方法が以下の二段階からなる。

1. 上で説明した様々な手がかりを用い、一般のトラブル表現を収集する。
2. トラブル表現をオブジェクトと組み合わせ、object-trouble ペアを獲得する。

4.1 トラブル表現の獲得

前述の様々な手がかりを用い、大規模 Web コーパスからトラブル表現の候補を抽出し、手がかり各種の有無を素性にし、教師あり学習である SVM を用い、実際にトラブル表現であるかどうかを判定する。学習に際し、抽出した語には正例が少ないことが分かった (約 6.5%)。しかし、精度が高い分類器を得るためには、学習データにある程度の数の正例サンプルが必要とされる。もっとバランスのとれた学習データ集合を構築すると同時にタグ付け作業の負担を減らすため以下の $Score$ 関数によって抽出結果をソートし、上位 2000 語と下位 2000 語で合計 4000 語を用意した。

$$Score(e) = \frac{f_{\text{LSPH}}(e) + f_{\text{DNV}}(e)}{f_{\text{LSPH}}(e) + f_{\text{DNV}}(e) + f_{\text{DAV}}(e)} \qquad (2)$$

ここで f_{LSPH}、f_{DNV} は表現 e の「LSPH で現れる頻度」、「DNV で現れる頻度」という肯定的手がかりの尺度で、f_{DAV} は e の「DAV で現れる頻度」を表す。この 4000 語を人手で判定して正解データを作成したところ、上位 2000 語には正例が 655 個、下位 2000 語には正例が 8 個となり、目的通り、ある程度バランスのとれた学習データが得られた。

学習では素性の値として各手がかりの有無 (0-1 の二値素性) を用いた。さらに、各手がかりの出現頻度を素性の値として学習することも行ったが、有意な精度向上は得られなかった。

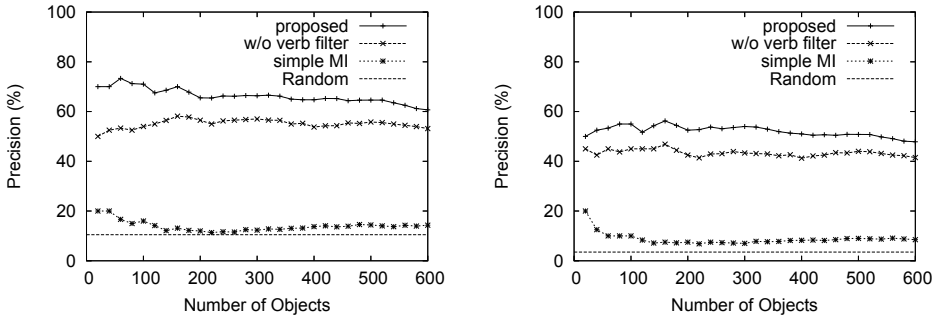
4.2 Object-trouble ペアの識別

第二段階では SVM によってトラブル表現として分類された語を様々なオブジェクトとペアにし、以下の尺度によりペアの関連を推定する。

$$I(e_o, e_t) = \frac{hit(\text{“}e_o\text{の }e_t\text{”})}{hit(\text{“}e_o\text{”})hit(\text{“}e_t\text{”})} \qquad (3)$$

この定義で e_o、e_t はそれぞれペアのモノとトラブルを指す表現である。$hit(e)$ は Yahoo! Japan の検索 API[*1] により得られた表現 e の hit count を表し、ここで e の一般的な頻度の推定値として扱う。

[*1] http://developer.yahoo.co.jp



(a) 評価者 3 人との一致率　　(b) 評価者 4 人との一致率

図 1　4 つの object-trouble ペアの獲得方法の精度

関数 I で得られるペア (e_o, e_t) のランキングは e_o が指すモノと e_t が指すトラブル表現の相互情報量に相当するものと考えられる。

最後はこのランキングの上位 N ペアに最終的なフィルターをかける。このフィルターの根拠になる仮定は前にも説明したが、あるトラブル表現 e_t があるモノ e_o の正常な用途の妨げになるのであれば、e_o と高共起頻度の動詞 v が存在し、その動詞 v を含む次のような DNV パターンが見つかると期待できるということである。

$$(\text{名詞 } e_t + \text{助詞:で}) \rightarrow \text{否定動詞 } v \qquad (4)$$

フィルタリングの手続きは I のランキングで上位 N ペアのオブジェクト e_o との共起頻度が高い動詞上位 M 語を調べ、コーパスにトラブル表現 e_t とそれらの動詞 v を含む DNV パターンが存在するかを確認する。それぞれ M 語の動詞と組み合わせ、そういったインスタンスが全く見つからない場合は候補ペアの (e_o, e_t) を最終出力から削除する。

5　評価実験

本研究で提案する方法を評価するために 4.6×10^8 文章の Web コーパスから LSPH・DNV パターンに現れる名詞句を抽出し、それらの LSPH・DNV・DAV パターンの頻度を記録した。低頻度のものを捨てた後に 53886 語が残った。

第一段階で教師あり学習により得られたトラブル表現上位 3000 語からランダムで 300 個をサンプルし、評価者 4 人に判定してもらった。評価者間の一致度を示す κ 統計量は 0.78 となり、これは十分な一致を示す。

まず、学習の結果が評価者 3 人と一致するケースについて報告する。トラブル表現のサンプル上位 50% の精度が 90% で、サンプル全体では 70% を超えた。ランダムサンプルから学習データに含まれた語を除くと精度が 80%(上位 40%) と 60%(全サンプル、220 個) まで下がった。評価者 4 人がトラブル表現として認める語のみ正解とすると、前述の精度が約 10% 低下する。本研究で提案する DNV・DAV 手がかりの有効性を示すために上の結果を LSPH パターンのみを用いる方法と比べた。同データから LSPH パターンで抽出した 2714 語からランダムで 200 個のサンプルを評価してもらった。その結果は 3 人の評価者との一致の場合で、精度が 52.5% で、4 人の場合は 43.5% であった。従って、DNV・DAV 手がかりでトラブル表現をより多く獲得できた上に、精度が LSPH のみを考慮する方法より優れていることが分かった。以上の実験から一般的にトラブルになりやすい表現の獲得に関して提案方法の有用性が明らかになった。

次は object-trouble ペアの識別の評価について報告する。合計 3.24GB の新聞記事[*2] で出現頻度 500 回以上の名詞からモノ・オブジェクトを指す語をランダムで 125 個を取り上げ、前段階で獲得したトラブル表現の上位 3000 語と組み合わせた。評価者 4 人に本研究のアプローチと共に以下の 3 つの方法から得られた結果それぞれ上位 600 個を評価してもらった。[*3]

ランダム　トラブル表現とオブジェクトをランダムで組み合わせた方法。(図 1：“Random”。)

DNV フィルター無し　提案のアプローチから最後の DNV フィルターを除いた方法。(図 1：“w/o verb

[*2] 読売 1987 年から 2001 年、毎日 1991 年から 1999 年、日経 1990 年から 2000 年。

[*3] ランダムの方は評価サンプルを 200 個に限った。

filter”。)

単純 MI 同上 DNV フィルターを除き、object-trouble ペアを以下の相互情報量のスコアでランキングした方法。(図 1：“simple MI”。)

$$MI(e_o, e_t) = \frac{hit(e_o, e_t)}{hit(e_0)hit(e_t)} \quad (5)$$

ここで “$hit(e)$” は Yahoo! Japan 検索 API により得られた表現 e の hit count を意味し、$hit(e_o, e_t)$ は e_o と e_t を両方含むページ数になる。

評価者にそれぞれのペア (e_o, e_t) についてはトラブル表現 e_t がオブジェクト e_o の用途や準備に対してのトラブルになるかどうかを判定してもらった。このタスクは評価者間の一致率を表す κ 統計量が 0.68 であった。図 1 は評価者 3 人 (グラフ (a)) と評価者 4 人 (グラフ (b)) について、それぞれの方法の精度を示す。本研究の方法は上位 150 ペアの精度で約 70% であり、テストサンプル全体の精度で約 60% である。「DNV フィルター無し」という方法との性能差が DNV パターンを用いた最終フィルタリングの有意性を裏付ける。さらに、「DNV フィルター無し」の結果がランダムよりはるかに精度が高いことが本アプローチのスコア関数 I の効果を明らかにするであろう。最後に、本実験では「単純 MI」の精度が意外に低かったことに驚いた。提案方法の精度との差は “e_o の e_t” というパターンに基づいたトラブル表現とオブジェクトの関連を推定する関数 I の効能を示す。

獲得したものの実例が図 2 に見れる。

ランキング - 評価者	object-trouble ペア
2 - 4	〈 薬 - 副作用 〉
3 - 4	〈 水槽 - 点検中 〉
15 - 3	〈 カテーテル - 痛さ 〉
18 - 0	〈 薬 - せい 〉
22 - 4	〈 フラッシュ - 光量不足 〉
18 - 0	〈 抗生物質 - アレルギー 〉
115 - 4	〈 モデム - 誤作動 〉
204 - 4	〈 センサー - 電池切れ 〉
404 - 1	〈 酒 - 臭い 〉

図 2 獲得した object-trouble ペアの例

6 まとめ

本研究で巨大な Web コーパスからモノとそのモノに対する様々なトラブルを自動的に獲得する方法について報告した。

今後の課題としては主に提案方法をスケール・アップする予定である。次期バージョンにもっと数多くのトラブル表現が獲得できるように、5 億文の日本語 Web コーパスを用い、利用に様々な制限がある検索 API への依存を無くし、かわりに対象表現の頻度等を効率よくコーパスから抽出する方法を考える。獲得の精度をさらに向上させるため、「e_t 防止」、「e_t 対策」、「e_t 予防」、「e_t 撲滅」等という特定の複合名詞を新しいトラブル表現の手がかりとして検討する。

参考文献

[1] Eugene Agichtein and Luis Gravano. *Snowball*: extracting relations from large plain-text collections. In *Proceeding of the 5th ACM International Conference on Digital Libraries*, pages 85–94, 2000.

[2] Maya Ando, Satoshi Sekine, and Shun Ishizaki. Automatic extraction of hyponyms from newspaper using lexicosyntactic patterns. *IPSJ SIG Notes*, 2003(98):77–82, 2003.

[3] Sergey Brin. Extracting patterns and relations from the world wide web. In *WebDB Workshop at 6th International Conference on Extending Database Technology, EDBT'98*, 1998.

[4] Ido Dagan, Oren Glickman, and Bernardo Magnini. The pascal recognising textual entailment challenge. In Joaquin Quinonero Candela, Ido Dagan, Bernardo Magnini, and Florence d'Alche Buc, editors, *MLCW*, volume 3944 of *Lecture Notes in Computer Science*, pages 177–190. Springer, 2005.

[5] M. A. Hearst. Automatic acquisition of hyponyms from large text corpora. In *Proc. of the 14th International Conference on Computational Linguistics*, pages 539–545, 1992.

[6] Takashi Inui, Kentaro Inui, and Yuji Matsumoto. Acquiring causal knowledge from text using the connective marker tame. *ACM Transactions on Asian Language Information Processing (TALIP)*, 4(4):435–474, 2005.

[7] Patrick Pantel and Marco Pennacchiotti. Espresso: leveraging generic patterns for automatically harvesting semantic relations. In *ACL '06: Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the ACL*, pages 113–120, Morristown, NJ, USA, 2006. Association for Computational Linguistics.

[8] Dmitry Zelenko, Chinatsu Aone, and Anthony Richardella. Kernel methods for relation extraction. *The Journal of Machine Learning Research.*, 3:1083–1106, 2003.