商品説明文からの属性・属性値の自動抽出

新里 圭司　　関根 聡
楽天技術研究所
{keiji.shinzato, satoshi.b.sekine}@.mail.rakuten.com

1　はじめに

近年，楽天，Amazon，eBay など多くのオンラインショッピングモールが様々な国でサービスを展開している．モールの利便性向上のためには販売されている商品データの構造化が重要であるが，多くの場合，モールへ出店している各店舗が手作業で行っており，コストの高いものとなっている．このため商品情報を自動で構造化する技術が求められている．モールで取り扱っている商品のバリエーションや，モールの国際化を考えると，構造化技術は言語非依存かつ，教師なし学習に基づく手法が望ましい．

注釈付きコーパス作成の手間を省略するため，Wikipedia や Freebase に代表される既存の知識ベース (KB) を活用した distant supervision [4] に基づく情報抽出手法が多く提案されている．しかし既存の KB は (1) モールのユーザが注目するような属性・属性値が登録されていない (Wikipedia)，(2) 英語に限定されている (Freebase)，ため商品情報を構造化する上で有効な知識源にならない．

本稿では，KB を商品ページから自動構築し，この KB を使って商品の属性・属性値を商品説明文から自動抽出する手法について述べる．具体的には，まず，商品ページに含まれる構造化された商品データ（表と箇条書き）を手がかりに KB を構築する．続いて，この KB を使って商品説明文に対して注釈付けを行いコーパスを自動作成する．最後に作成したコーパスを基に属性値の抽出モデルを学習する．構造化された商品データは店舗によって記述されているためある程度信頼の置けるデータとして見なせるが，一方で，標準的な書式がないため店舗によって表記にバラつきがある．この問題を解決するため，本手法では属性値を手がかりに同義と見なせる属性同士を自動でまとめあげる．また自動で作成したコーパスには，注釈付けの誤りや漏れが含まれる．そこで，KB のエントリ (<*attr*, *value*>) の頻度及びエントリから生成されたパターンを使って問題となる注釈付けを含む文を削除し，コーパスの品質向上を図る．

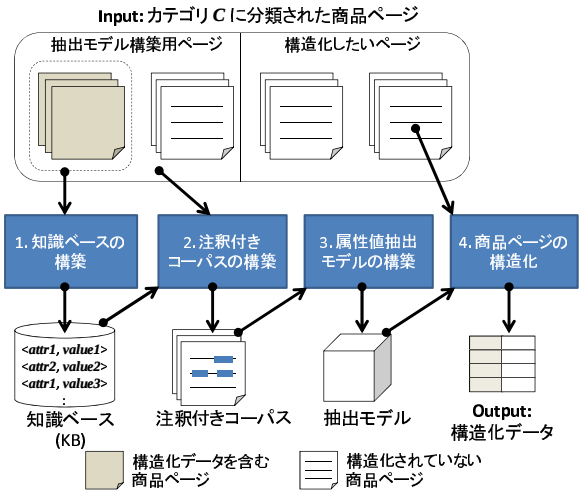

図 1: 提案手法の概要

2　関連研究

本研究は Ghani ら [1] の研究と類似している．Ghani らは人手で作成した KB を利用して，注釈付きコーパスを自動構築し，このコーパスを用いて属性値の抽出モデルを学習している．本研究は，KB を商品ページ中の構造化されたデータを利用して自動構築している点が異なる．

商品に関する KB の構築という観点では Yoshinaga らの手法 [7] と近い．Yoshinaga らは属性の表記のゆれの問題は扱っていないが，本手法では表記ゆれの問題解決も試みる．属性の表記ゆれ問題の解決は Mauge ら [3] も試みているが，Mauge らは教師データを必要とするのに対し，本手法では必要としない．

3　商品データ

本研究では楽天市場の商品データ (HTML タグ付) を利用する．データは約 1 億件の商品からなっており，各商品は約 4 万ある商品カテゴリのいずれか 1 つに分類されている．分類は楽天で商品を販売している店舗により行われており，誤ったカテゴリに分類された商品も存在する．

仮説 1: 表のヘッダーに書かれやすい語は属性である．
仮説 2: 属性 a, b が同一の構造化データに出現しておらず，a, b が店舗頻度の高い同一の属性値をとる場合，a, b は同義である．
仮説 3: 構造化データ中で店舗頻度の低い属性値は，非構造化データ中においても店舗頻度が低い．

図 2: 本手法で用いる仮説

P1: `<T(H|D)>[ATTR]</T(H|D)><TD>[ANY]</TD>`
P2: `[P][ATTR][S][ANY][P]`　　**P3:** `[P][ATTR][ANY][P]`
P4: `[ATTR][S][ANY][ATTR][S]`

図 3: 属性値抽出のための正規表現．[ATTR] は属性，[ANY] は任意の文字列，[P] は prefix と開き括弧，[S] は suffix と閉じ括弧をそれぞれ表す．prefix，suffix，括弧は吉永ら [7] で定義されたものと同じ．

4 提案手法

提案手法の概要を図 1 に，用いる仮説を図 2 に示す．以下，図中の Step 1–3 について述べる．Step 4 については，Step 3 で構築されたモデルを与えられた商品ページに適用するだけであるため省略する．

4.1 知識ベースの構築

4.1.1 属性・属性値の獲得

前述したように商品ページに含まれる表および箇条書きを手がかりに KB を構築する．その際，図 2 の仮説 1 に従い，正規表現パターン<TH>.+?</TH>を使って属性を獲得する (<TH>は表のヘッダーを表す HTML タグ)．獲得された属性のうち**保存方法, その他, 商品説明, 広告文責, 特徴, 仕様**は適切な属性と見なせないため除く．

続いて属性・属性値の組を抽出するため，図 3 に示した正規表現パターンを商品ページに適用する．[ANY] にマッチした表現を [ATTR] に対応する属性の値として抽出する．なお **P4** においては，最初に出現した [ATTR] の値とする．抽出された属性およびその値を KB に登録する際，当該データを構造化データ中に記述した店舗数も合わせて登録する．本稿ではこの店舗数のことを**店舗頻度**と呼ぶ．

4.1.2 属性の同義語の自動認識

前節で構築した KB は属性の表記にゆれがある．これは商品データを店舗が記述するための標準的な方法 (規則) がないためである．例えば,「イタリア」「フランス」はワインカテゴリにおいて「生産地」であるが，店舗 m_1 は「生産地」，店舗 m_2 は「生産国」として記述することがある．表記ゆれの問題を解決せずに KB を注釈付けに利用すると，注釈に一貫性のないコーパスが得られ問題となる．

表記ゆれを解決するため仮説 2(図 2) を用いる．具体的には，まず，店舗頻度が N を超える KB のエントリを対象に，同じ属性値を持つ属性のベクトル (a_1，a_2) を生成する．そして，属性 a_1，a_2 が同一の構造化データに含まれているかどうかチェックし，含まれていなければそれらを同義語と見なす．N は $N = \max(2, M_S/100)$ で求まる値であり，M_S は対象カテゴリにおいて構造化データを提供している店舗数を表す．この処理により，例えば v_1 =(生産国, 生産地), v_2 =(ぶどう品種, 品種) が得られる．得られた属性のベクトルの集合 ($\{v_1, v_2, \cdots\}$) を S_{attr} で表す．

表 1: ワインカテゴリ用 KB の例

属性	同義語	値の異なり数	属性値の例 (括弧内の数字は店舗頻度)
容量	内容量	159	750ML[147], 720ML [64],375ML [49], 500ML [41],1500ML [22], 360ML [15],200ML [13]
品種	ぶどう品種, ブドウ品種, 使用品種, 葡萄品種	1,578	シャルドネ [59], メルロー [36], リースリング [29], シラー [29], グルナッシュ[22], サンジョヴェーゼ [20], メルロ [20]
タイプ	—	153	辛口 [34], 赤 [24], 白 [23], フルボディ [23], やや甘口 [15], 甘口 [14], ライトボディ [12]

続いて，属性ベクトルの集合 S_{attr} の中で類似度の高い属性のベクトル同士をマージする．例えば，(地域, 生産国, 生産地) は，(地域, 生産国)，(生産国, 生産地) から得られる．ベクトル v_i, v_j 間の類似度はコサイン尺度 $sim(v_i, v_j) = v_i \cdot v_j / |v_i||v_j|$ で求める．マージ処理は類似度の最大値が 0.5 を下回るまで繰り返し行う．

表 1 にワインカテゴリに対して構築された KB の例を示す．表より属性「品種」は 4 種類の異なる属性と同義であることがわかる．

4.2 注釈付きコーパスの構築

4.2.1 注釈付け

商品ページ中のテキストを，句点，括弧，ブロック要素タグを手がかりに文に分割し，各文を形態素解析する．次に，形態素列に適合する最長の属性値を持つエントリを KB から選択し，属性値に対応する属性を注釈付けする．適合するエントリが複数ある場合は店舗頻度が最大のものを選択する．

4.2.2 注釈誤りの除去

店舗頻度の低い KB のエントリは誤りの可能性が高く，このような誤ったエントリが頻繁に注釈付けされるとコーパスの質が低下する．そこで，仮説 3 に従い誤ったエントリに基づいて注釈付けされた表現を含む文を検出し，コーパスから削除する．誤りを検出するため以下のスコアを用いる．

$$Score(v) = \frac{MF_D(v)/N_M}{MF_S(v)/M_S}$$

ここで $MF_D(v)$ と $MF_S(v)$ は，商品説明文および構造化データ中での属性値 v の店舗頻度を表す．また N_M

MP1:[NUMBER] [ML], **MP2:**[シャトー] [ANY],
MP3:[LOCATION] [ANY] [LOCATION] [市],
MP4:[カベルネ] [ANY] [%], **MP5:**[ANY] [地方]

図 4: ワイン用 KB から生成された形態素パターンの例．[ANY] は任意の 1 - 3 形態素，[LOCATION] は品詞が地名，[NUMBER] は数詞の形態素にマッチする．

基本素性: 単語の表層形，基本形，品詞，前 2 文字 (e.g., **シャ**トー)，後 2 文字 (e.g., シャ**トー**)，単語内の文字の種類 (平仮名，カタカナ，漢字，アルファベット，数字，その他).
文脈素性: 前後 3 単語の基本素性

図 5: 素性

は対象カテゴリにおいて商品を販売している店舗の数を表す．上記のスコアは構造化データおよび商品説明文（非構造化データ）のどちらにおいても店舗頻度の低い属性値に対して低い値を与える．本研究では，上記のスコアが 30 以上の属性値 v を誤りと見なし，この属性値を含む文をコーパスから除く．

4.2.3 注釈漏れの除去

KB のカバレッジが低いため，構築されたコーパスには注釈付けの**漏れ**が含まれる．例えば，文「シャトータルボといえば，言わずと知れた**<地域>サン・ジュリアン</地域>**を代表するシャトー.」において生産者「シャトータルボ」が注釈されない，という場合がある．この場合,「シャトータルボ」はモデル構築時に負例として扱われてしまうため問題となる．

このような注釈付けの漏れを含む文をコーパスから削除するため，形態素パターンを KB 内のエントリから自動生成する (図 4 参照)．具体的には，KB 内の属性値を形態素解析し，解析結果に対して PrefixSpan アルゴリズム [5] を適用し形態素パターンを導出する．パターンにマッチする形態素列を含む文は，注釈付けの漏れがあると判断しコーパスから除く．

4.3 属性値抽出モデルの構築

Step 2 で構築した注釈付きコーパスを用い，属性値抽出のためのモデルを Conditional Random Field (CRF)[2] で学習する．学習に用いた素性を図 5 に示す．CRF の実装には，CRFsuite[1]を用い，学習時のパラメータはツールのデフォルトの値に従った．またタグの表現には Start/End 法 [6] を利用した．

5 実験

5.1 評価用コーパスの作成

表 2 に示す 16 カテゴリから抽出した 3,740 商品ページに対して人手で注釈付けを行い，評価用コーパスを作成した．まず各カテゴリから出品数の多い店舗上位

表 2: 評価に用いたカテゴリおよび属性

カテゴリ	属性 ($^{\#}$: 削除, *: マージ)
ワイン	容量, 品種, タイプ, 産地, アルコール度数, 原産国 *, 生産者, 原材料, アルコール分
ミネラル水	容量, 原材料, 賞味期限, ナトリウム, 産地, カルシウム, マグネシウム, 採水地
男性用 T シャツ	サイズ, 素材, 色, 着丈 *, 身幅 *, M$^{\#}$, 肩幅 *, L$^{\#}$
女性用カットソー	サイズ, 材質, 着丈 *, 色, 肩幅 *, 袖丈 *, バスト *, 身幅 *
ショルダーバッグ	材質, サイズ, 色, 重さ, 製造国, 付属品, ショルダー *, マチ *
携帯電話ケース	材質, サイズ, 重さ, 色, 付属品, 製造国, 対応機種, 商品サイズ *
プリンタインク	容量, サイズ, カラー, 重量, 色 *, 適応機種, 材質, 製造国
シャンプー	容量, メーカー, 製造国, 成分, 商品名, 区分, サイズ, 重量
化粧水	容量, メーカー, 成分, 製造国, 区分, サイズ, 原産国 *, 重量
釣りルアー	タイプ, 重さ, フック, アクション, リング, レンジ, 全長, メーカー
ゴルフボール	コア, サイズ, カバー, 材質, 重さ, 原産国, ディンプル形状, 色
TV ゲーム	サイズ, 重さ, 材質, 付属品, 製造国, 色, 対応機種, ケーブル長
フィギュア	サイズ, 材質, メーカー, 対象年齢, パッケージサイズ *, ジャンル, 発売日
自動車ライト	サイズ, 色温度, 色, 材質, 重量, 適合車種, 製造国, 品番
キャットフード	内容量, 原産国, 粗繊維, 粗脂肪, 粗灰分, 水分, 粗タンパク質, 重量 *
柔軟剤	内容量, 原産国, 成分, サイズ, 用途, 名称, 重量, 液性

表 3: コーパスの統計．$p^{\#}$, $s^{\#}$, $v^{\#}$ は注釈付けされたページ数，文数，属性値数をそれぞれ表す．

カテゴリ	コーパスの統計量					
	人手			自動		
	$p^{\#}$	$s^{\#}$	$v^{\#}$	$p^{\#}$	$s^{\#}$	$v^{\#}$
ワイン	280	1,770	2,856	20,541	25,407	40,664
ミネラル水	278	1,325	1,633	2,907	7,778	10,348
男性用Tシャツ	256	2,553	5,016	15,151	18,249	41,448
女性用カットソー	276	1,750	3,341	13,164	21,781	41,719
ショルダーバッグ	282	1,650	2,989	13,874	18,883	30,588
携帯電話ケース	247	1,688	2,914	7,152	14,106	21,190
プリンタインク	265	1,176	3,071	8,493	13,598	42,313
シャンプー	232	1,259	2,354	18,669	30,263	52,193
化粧水	266	1,710	3,446	13,889	23,517	45,447
釣りルアー	91	334	499	9,378	16,798	19,493
ゴルフボール	158	544	703	1,122	2,129	2,736
TV ゲーム	209	779	1029	19,434	29,566	34,278
フィギュア	254	1,130	1,522	16,666	28,528	39,068
自動車ライト	268	1,360	2,271	8,125	12,911	18,610
キャットフード	86	269	437	4,924	7,392	8,687
柔軟剤	272	1,508	2,586	2,330	5,397	8,620
合計	3,720	20,805	36,667	175,819	276,303	457,402

300 件を抜き出し，各店舗の商品ページをランダムに 1 件抜き出した．各ページからはタイトルおよび商品説明文を抜き出し，注釈付けの対象とした．

注釈付けは，4.1.1 節の方法で各カテゴリについて獲得した属性名のうち，店舗頻度が高い上位 8 件を対象に行った．8 件の属性のうち，カテゴリに対して不適切なものは人手で削除し，まとめあげられていない同義語は人手でまとめた．削除およびマージされた属性は，表 2 において，$^{\#}$，* を記してある．5.3 節で述べる評価実験では，これらの属性に関して抽出された属性値は評価対象から外している．

アノテータには，各属性の値として適切な表現に対してその属性名を注釈付けするよう依頼した．また，注釈付けの際，1 商品ページで複数商品が販売されているものは無視するよう指示した．

作成した評価用コーパスの統計データを表 3 に示す．作成には 1 人のアノテータで 2ヶ月要した．

[1]http://www.chokkan.org/software/crfsuite/

表 4: 自動構築された KB の精度

店舗頻度	ワイン		シャンプー	
	エントリ数	Prec. (%)	エントリ数	Prec. (%)
≥ 1 (All)	3,940	78.6	6,798	75.0
≥ 2	751	93.9	2,307	91.7
≥ 3	384	96.9	1,543	94.5
≥ 5	215	97.7	931	94.8

表 5: システムの精度 (P，R，F 値はマクロ平均)

手法	P (%)	R (%)	F 値
(a) KB エントリの単純なマッチング	38.57	42.23	40.32
(b) 抽出モデル (コーパスのフィルタ無)	39.27	49.00	43.60
(c) 抽出モデル (提案手法)	**42.14**	**51.78**	**46.46**

5.2 知識ベースの評価

5.2.1 獲得された属性の評価

4.1.1 節の方法で獲得された属性が当該カテゴリに対して適切であるかどうか，被験者 1 名を使って評価した．評価の対象は 16 カテゴリに対して獲得された属性 699 件である．評価の結果「適切」と判定された属性の割合は 0.820 であった.「不適切」と判定された属性の多くは，複雑な構造を持つ表データ，店舗によって誤ったカテゴリが割り当てられた商品ページ中の表データから抽出されたものであった．

5.2.2 属性の同義語の評価

次に属性の同義語の評価を行った．具体的には，各カテゴリにおいて獲得された適切な属性を,「同義かどうか」という観点でクラスタリングするよう被験者に依頼し，このデータを使って自動生成された同義語集合の *purity* および *inverse-purity* を計算した．

実験の結果，*purity* は 0.935，*inverse-purity* は 0.848 であった．*purity* の値は 1 に近いことから，同義語として自動認識された属性同士は多くの場合正解であったことがわかる．一方で *inverse-purity* の値は 0.848 であったため，手法の網羅性という観点では改善が必要なことがわかる．

5.2.3 知識ベースのエントリの評価

ワインとシャンプーカテゴリについて，KB エントリの評価を行った．カテゴリ C について構築した KB のエントリ<*attr*, *value*>を評価する際，文「*value* は C の (*attr* or S^1_{attr} or S^2_{attr} or ... or S^n_{attr}) を表す表現である」が被験者にとって自然である場合,「適切」と判定するよう指示した．ここで S^n_{attr} は属性 *attr* の n 番目の同義語である．

評価結果を表 4 に示す．表より店舗頻度が高いエントリ程適切であることがわかる．特に店舗頻度が 2 以上のエントリでは 90%以上が適切であることがわかる．

5.3 属性値抽出モデルの評価

各カテゴリに対して構築した抽出モデルの評価を行った．モデル構築の際，RAM の容量が不足したため，学習に用いる文をサンプリングした．具体的には，KB のエントリを単純にマッチングすることでコーパスを一度作成し，このコーパスから注釈付けされた文を 10 万文ランダムに抽出した．この 10 万文に対し，4.2 節で述べた注釈付けの誤り及び漏れを検出するフィルタを適用し，最終的に残った文から抽出モデルを学習した．構築したコーパスの統計データを表 3 に示す．

実験結果を表 5 に示す．比較のため，KB エントリの単純なマッチングを行った場合，フィルタを用いずに構築したコーパスを使って抽出モデルを学習した場合の性能も示す．手法 (c) は手法 (b) で構築したモデルよりも F 値で約 3 ポイント高い．これはフィルタが有効に働いていることを示唆している．手法 (c) における false-positive/negative の主なものは KB の精度およびカバレッジによるものであった．品質の高い KB の構築は今後の課題である．

手法 (b) は (a) に比べ 3 ポイント以上 F 値が高い．これは抽出モデルが属性値の文脈及び属性値を構成する語のパターンを学習した結果であると考えられる．

6 おわりに

本稿では，構造化されていない商品ページから，商品の構造化データを自動で抽出する手法について述べた．自動注釈付けに用いる KB の精度およびカバレッジが，その後の抽出モデルの性能に影響することがわかったため，今後は高品質な KB の構築方法に注力したい．

参考文献

[1] Rayid Ghani, Katharina Probst, Yan Liu, Marko Krema, and Andrew Fano. Text mining for product attribute extraction. *ACM SIGKDD Explorations Newsletter*, 8(1):41–48, 2006.

[2] John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of ICML2001*, pages 282–289, 2001.

[3] Karin Mauge, Khash Rohanimanesh, and Jean-David Ruvini. Structuring e-commerce inventory. In *Proceedings of ACL2012*, pages 805–814, 2012.

[4] Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. Distant supervision for relation extraction without labeled data. In *Proceedings of ACL-IJCNLP2009*, pages 1003–1011, 2009.

[5] Jian Pei, Jiawei Han, Behzad Mortazavi-Asl, Helen Pinto, Qiming Chen, Umeshwar Dayal, and Mei-Chun Hsu. Prefixspan: Mining sequential patterns efficiently by prefix-projected pattern growth. In *Proceedings of ICDE2001*, pages 215–224, 2001.

[6] Satoshi Sekine, Ralph Grishman, and Hiroyuki Shinnou. A decision tree method for finding and classifying names in japanese texts. In *In Proceedings of the Sixth Workshop on Very Large Corpora*, 1998.

[7] Naoki Yoshinaga and Kentaro Torisawa. Finding specification pages according to attributes. In *Proceedings of WWW2006*, pages 1021–1022, 2006.