Domain Independent Model for Product Attribute Extraction from User Reviews using Wikipedia

Sudheer Kovelamudi[+] **Sethu Ramalingam**[*] **Arpit Sood**[+] **Vasudeva Varma**[+]

[+]International Institute of Information Technology, Hyderabad, India
sudheer.k@research.iiit.ac.in, arpit.soodug08@students.iiit.ac.in, vv@iiit.ac.in
[*]24/7 Customer Ilabs, India
sethu.s@247customer.com

Abstract

The world of E-commerce is expanding, posing a large arena of products, their descriptions, customer and professional reviews that are pertinent to them. Most of the product attribute extraction techniques in literature work on structured descriptions using several text analysis tools. However, attributes in these descriptions are limited compared to those in customer reviews of a product, where users discuss deeper and more specific attributes. In this paper, we propose a novel supervised domain independent model for product attribute extraction from user reviews. The user generated content contains unstructured and semi-structured text where conventional language grammar dependent tools like parts-of-speech taggers, named entity recognizers, parsers do not perform at expected levels. We used Wikipedia and Web to identify product attributes from customer reviews and achieved $F_1 score$ of 0.73.

1 Introduction

The online retail market is growing immense, offering millions of products for customers. The products are generally described in terms of a few set of attributes. Such product attributes are mined from the descriptions to represent the product in a structured manner.

Often descriptions deal with generic attributes. For example, specific attributes like power consumption, pulsator, load, spin-dry effectiveness, noise, water usage, water leakage, etc for a product like washing machine cannot be correctly found in descriptions. On the other hand, customers express their opinions in the form of reviews. The opinions expressed are in terms of attributes they like and dislike but not always in terms of those attributes that are provided by the retailer for that particular product. Hence mining the attributes about which the customers discuss can be really helpful for retailers as well as for other customers.

Mining product attributes from customer reviews can lead retailers to fetch and group other products that are having similar specific attributes and forecast more precisely. Hence many retailers are trying to enrich their product knowledge bases with these domain specific and product specific attributes. Attribute extraction from reviews is also useful in tasks like review summarization, product rating, sales agent assessment, opinion mining of reviews, product recommendation systems, customer relationship management, customer satisfaction analysis, customer profiling, etc.

On the customers' side, they are prone to seek the opinions of other customers who actually used the product or bought it from a retailer website. They ask for unbiased evaluation of a product by leveraging information from multiple reviews, although each individual review can be subjective in nature. Therefore a person is more interested to read a featured review than overall reviews like "the product is really great, awesome!" or "this is the greatest product I have ever seen!!!" or simply the product rating.

Consider an unstructured customer review on the product

LG Electronics Flatron L1920P monitor:

> Excellent *picture* quality.. videoz are in *HD*.. no complaintz from me. Never had any trouble with gamez.. Paid WAAAAY to much for it at the time th0.. it sellz now fer like a third the *price* I paid.. heheh.. oh well....the fact that I didn't wait a year er so to buy a bigger *model* for half the price.. most likely from a different *store*.. ..not namin any namez th0.. *cough*BBHOSEDMe*cough*

The italicized terms are some product attributes discussed in this review. Our aim is to extract such attributes automatically. The reviews act as good sources in supplying such product specific attributes.

Mining attributes from customer reviews is a challenging task as they mostly comprise of user generated content. The text in such user generated content is low in natural language grammar, structure, formality. It often hinders the performance of natural language processing tools like parts of speech tagging, parsing and named entity recognition.

By this motivation, we have designed a novel framework that can extract attributes of a product with out making use of any natural language tools but treating the text as 'Bag Of words' and using the knowledge of Wikipedia.

2 Related Work

A good amount of research had been put into product attribute extraction in recent years. But the focus was laid in extraction of attributes from product descriptions and a little was done in extracting the same or more specific attributes from user reviews. Much of the existing work focuses on whole review classification and overall opinion extraction.

Work related to word order occurrences where product attributes are believed to exist as noun phrases was already contributed (Justeson and Katz, 1995; Daille, 1996). But it (Hu and Liu, 2006; Hu and Liu, 2004; Liu et al., 2005) was shown that using noun phrases tend to produce too many non-terms (low precision), while using reoccurring phrases misses many low frequency

Proceedings of the 5th International Joint Conference on Natural Language Processing, pages 1408–1412,
Chiang Mai, Thailand, November 8 – 13, 2011.

terms, terms with variations, and terms with only one word. Their work presents the identification of product attributes with the help of Parts-Of-Speech(POS) tags and the occurrence of adjectives. But in most of the cases when free format reviews are considered, the POS taggers do not function at the expected level as grammar is not guaranteed in user generated text.

Efforts like training noun phrase recognizer model (Raju et al., 2009) to extract attributes from product descriptions worked well on structured text, but when tested did not work on unstructured text and long reviews.

The major extraction problem that has been studied extensively is the named entity extraction. We tried extracting product attributes from a set of reviews which consist of incomplete sentences and short phrases (using the technique given by (Liu et al., 2005)), but the results are not consistent. The reason we believe is that, in most of the cases product attribute terms do not have indicators (like beginning with capitalization of letters) as in the scenario of customer reviews which consequently result in named entity recognition failure.

3 Attribute extraction

For any given product, our approach to attribute extraction involves:

1. Collection of customer reviews of the given product.
2. Filter out stop words.
3. Compute features that we have defined, for the remaining words.
4. Identification of possible attribute words using classification model trained on these features.

Support vector machines (SVMs), are a set of related supervised learning methods for classification and regression analysis which are used to facilitate our model. The features on which our system has been trained are explained in the following sections.

3.1 Most Frequent Items-MFI

Words related to topics that are discussed more occur at high frequencies in any given text. In general people discuss about the attributes of a product in their reviews frequently. The 'Most Frequent Items' feature boosts the importance of attribute words by their frequency of occurrence in customer reviews.

The set of words $\{z_1, z_2, z_3, ...z_m\}$ used for this feature are obtained from customer reviews of a given product after stop word removal is done. For any word z_i the 'Most Frequent Items' feature is computed by

$$MFI(z_i) = \frac{Freq(z_i)}{\sum_{j=1}^{m} Freq(z_j)}$$

$Freq(z_i)$ gives total number of occurrences of z_i in reviews of a given product.

3.2 Context Relation using Wikipedia - CR

To understand a context or to identify a context, we need the set of keywords that portray the context. So, we assume that any context C can be expressed as $C = \{t_1, t_2, t_3, ...t_n\}$ where 't_i' are the related keywords dealt in C.

The product forms the context in customer reviews. People talk about the product and its attributes in their reviews. Its attributes and other highly related things belong to the set of keywords of the context.

The CR feature is about identifying the list of related keywords mentioned in customer reviews that can be found in Wikipedia. We start with identifying all words that have been discussed in reviews of a given product in Wikipedia and then proceed with calculating the most semantically related words among them.

When we make judgments about semantic relatedness between any two words, we draw huge amount of background knowledge about the concepts that these words represent. Hence, any trial to state the semantic relatedness between different words automatically also needs to do the same. One can use hand-crafted lexical structures like thesauri and taxonomies, or statistical analysis of large corpora to process the semantic decisions automatically (Milne, 2007). The limiting factors of such techniques when carried across domains are the background knowledge, precision, scalability and scope. With more than a 18 million articles and thousands of volunteers all over the world, Wikipedia which is a growing massive repository of knowledge, is the best alternative when targeted by such limitations.

We explore Wikipedia's link structure, category structure, article titles, and page types from the static and latest pages-articles xml dump[1] of Wikipedia. We only need Wikipedia's structure rather than it's full textual content. We have created SQL database, tables to store and access the page titles and articles fast, which has been suggested and explored already (Milne and Witten, 2009). We map a word in customer reviews to a Wikipedia article if the word is contained in that Wikipedia article title. We call such words as *Wikipedia words* and if cannot be mapped, we refer them as *Non-Wikipedia words* in later sections of this paper. A word can be mapped to all its homonyms in Wikipedia. For instance the word 'bank' can refer to 'river bank' or a 'savings bank' in Wikipedia. To disambiguate and identify the correct possible article mappings for a given word, we need to first disambiguate words which may possibly contain mappings in more than one domain. To address this, we used a method (Milne and Witten, 2009) where articles for unambiguous words are used to disambiguate the ambiguous words.

Computing semantic relatedness between two words that are mapped to Wikipedia, is equal to finding the semantic relatedness between articles in Wikipedia to which these words refer. And to do this, the best known way is to compute the relation from the links to these articles in Wikipedia (Medelyan et al., 2008; Milne, 2007).
The relation between two Wikipedia articles x and y is given by

$$Relation_{x,y} = 1 - \frac{max(log|A|, log|B|) - log|A \cap B|}{T - min(log|A|, log|B|)}$$

Here A and B are the set of articles which link to the articles x and y respectively, T is the total number of Wikipedia articles, $A \cap B$ is their overlap. Thus for every *Wikipedia word*, we find the semantic relatedness to all other such words. Context relatedness feature (CR) of a word is computed as the sum of its similarity

[1] `http://dumps.wikimedia.org/enwiki/`

scores with all other such words in the context which is then normalized by the total number of such words. Therefore for a Wikipedia words set $\{x_1, x_2, x_3, ...x_k\}$, semantic relatedness of x_i to the context is given by

$$CR_{x_i} = \frac{\sum_{\substack{j=1 \\ j \neq i}}^{K} Relation_{x_i,x_j}}{k}$$

The applicability of CR feature is justified in terms of high scalability and the ever growing knowledge of Wikipedia.

For *Non-Wikipedia words* $\{y_1, y_2, y_3, ...y_l\}$ in the product reviews, the CR feature is modified as the average of all CR feature values for $Wikipedia\ words$, from reviews of that particular product. Hence the CR value for any non-Wikipedia word y_i is uniformly given as

$$CR_{y_i} = \frac{\sum_{j=1}^{k} CR_{x_j}}{k}$$

where x_j is a Wikipedia word.

3.3 Role of surrounding window - SW

We have taken into account the surrounding text of 't' Wikipedia words to the left and right of a given Wikipedia word to examine its role in identifying an attribute. As some topics arise and eventually diminish in a small window of discussion, the situation motivates us to consider the relation with the surrounding text as a classification feature in identifying product attributes.

This feature can help in identifying sub-attributes (attributes of attributes). The sub-attributes may not seem related when overall context is considered, but they are relevant when limited contexts in which they occur are considered.

Suppose if there are p instances of Wikipedia word x_i in the reviews. The relation of x_i with the surrounding text is computed as

$$SW_{x_i} = \frac{\sum_{\substack{j=-t \\ j \neq i}}^{t} Relation_{x_i,x_j}}{(k)(N)(p)}$$

Where N is the total number of words in customer reviews of a given product. The window length t is arbitrarily taken as $\frac{N}{20}$. "-t" means t words to the left of x_i and vice-versa.

The SW feature for the non-Wikipedia words is uniformly given as average of all SW feature values of Wikipedia words from reviews.

$$SW_{y_i} = \frac{\sum_{j=1}^{k} SW_{x_j}}{k}$$

3.4 Web search engine reference-WR

As there are words that cannot be mapped to Wikipedia, we may loose a few trivial attributes in the candidate selection stage. To boost such words we use knowledge on the Web. The WR feature measures the association of a particular word from customer reviews of a product with that product on the Internet.

We have used Bing search API[2] to compute WR for a word. WR value for a word z_i is given by

$$WR_{z_i} = \frac{Res(z_i, P)}{S_N}$$

$Res(Z_i, P)$ is the number of instances where the word z_i and the product name P both occur within the text snippets given as search results by the search engine. This frequency is normalized by the total number of search results S_N that are taken into account.

4 Experiments and Evaluation

We have trained our system with the above features using SVM. We have evaluated our system against two popular datasets of reviews, the Reviews-9-products dataset (Ding et al., 2008) and the Customer Reviews dataset (Hu and Liu, 2004). These datasets have been used for the opinion mining tasks and referred by several other publications[3]. They were annotated manually in terms of product attributes. These annotations consists of trivial words, terminologies, and concepts. The datasets contain customer reviews of products from different domains of Amazon[4].

Experiments are carried out at two levels. First, crucial features are tested to know their respective performance, and then the complete combination of features is tested. To train our model we used $Reviews-9-products$ dataset and for testing $CustomerReviews$ dataset is used. Similarly we have also done testing on $Reviews-9-products$ dataset by generating the training data from $CustomerReviews$ dataset.

We have considered MFI feature as baseline for this approach. The reason is that MFI is intuitive due to the fact that people when discussing about a product mention the attributes a good number of times in their reviews.

$$Precision = \frac{No.\ of\ Attributes\ Identified\ correctly}{No.\ of\ words\ Identified\ as\ Attributes}$$

and the recall is given by

$$Recall = \frac{No.\ of\ Attributes\ Identified\ correctly}{No.\ of\ Attributes\ Actually\ Annotated}$$

4.1 Product attribute extraction using Wikipedia

When we have tested our Wikipedia based features CR, SW along with the baseline feature MFI, we encountered a low recall but a good average precision of approximately 88%. The reason behind this low recall is that trivial words and some verbs cannot be mapped to Wikipedia. For example, for the *DiaperChamp* product listed in Table 1 the annotated attributes like $bang-for-the-buck, deal, looking, filelimit, cost-effective, works, pull, assemble, costlier, clean, safer$, *etc.*, cannot be correctly linked to the articles of

[2] http://msdn.microsoft.com/en-us/library/dd251072.aspx

[3] http://www.cs.uic.edu/\~liub/FBS/sentiment-analysis.html

[4] http://www.amazon.com/

Table 1: Performances of different combinations of features

Product Name	Annotated Attributes	CR,SW,MFI		CR,SW,MFI,WR	
		Candidates Selected	Attributes Identified	Candidates Selected	Attributes Identified
Diaper Champ	68	16	14	57	45
Canon G3	106	30	25	93	70
Hitachi router	82	14	11	79	66
Canon S100	99	26	23	91	73
Nokia 6600	147	48	44	112	85
MicroMP3	196	41	35	133	102
Nikon coolpix 4300	76	16	13	54	46
ipod	92	23	10	85	66
Creative Labs Nomad Jukebox Zen Xtra 40GB	186	47	43	157	122
norton	107	24	23	94	73
Linksys Router	85	24	18	79	52
Apex AD2600 Progressive-scan DVD player	115	24	19	90	79
Canon PowerShot SD500	70	13	12	63	52
Nokia 6610	111	35	31	92	74

Table 2: Overall scores

Feature combination	Recall	Precision	F-score
Baseline(MFI)	0.112	0.603	0.189
CR, SW, MFI	0.202	**0.878**	0.328
CR, SW, MFI, WR	**0.666**	0.802	**0.727**

Wikipedia. To rule out such discrepancies we can use an ontology like Wordnet. But it adds a lot of noise. The statistics of the identified attributes from both datasets are shown in Table 1 and their collective precision, recall and f-score values are given in Table 2.

4.2 Product attribute extraction using Wikipedia & Web

The web based feature WR when combined with other features increased recall of our system.

We can clearly see that the combination of all the four features which include Wikipedia based features and other frequency, web based features has performed the best in terms of f-score. The increase in recall is due to gain in knowledge using WR. The fall in precision can be explained by the boosting of insignificant words in search results.

In Table 1, the given products *Diaper Champ* and *ipod* belong to the most divergent domains. For *Diaper Champ* our model identified 45 out of 68 annotated attributes where as for *ipod*, it identified 66 out of 92 annotated attributes. Similarly for the product *Apex AD2600 Progressive-scan DVD player*, it identified 79 out of 115 attributes. This shows that the recall is approximately equal across the products which is an evidence that the model does not depend on the domain of a product.

5 Conclusion and Future Work

In this paper, we presented a domain independent approach for automatic discovery of product attributes from user reviews. Our work has highlighted the possibility of providing an incremental learning capability for an extraction system. The performance scores of our system show that it is a good design to apply Wikipedia to carve out product attributes from customer reviews. Our contribution is in leveraging information and in getting assistance from greater knowledge sources like Wikipedia and world wide web when doing tasks across domains while discarding all the help from language tools.

In this work we have trained and tested our system over products that belong to different domains but interestingly found it works uniform for all the products. In future we want to test our model extensively across domains and explore new methodologies for generic attribute extraction. We would like to extend the model for sentiment analysis on product attributes mined from reviews. Our future research work also include testing our system using other machine learning techniques (like CRF) and to consider more baselines for evaluation. As we did not make use of any natural language processing tools, this work can be extended to any other language with little changes in the preprocessing stage.

References

B. Daille. 1996. Study and implementation of combined techniques for automatic extraction of terminology. *The Balancing Act: Combining Symbolic and Statistical Approaches to Language*, 1:49–66.

Xiaowen Ding, Bing Liu, and Philip S. Yu. 2008. A holistic lexicon-based approach to opinion mining. In *Proceedings of the international conference on Web search and web data mining*, WSDM '08, pages 231–240, New York, NY, USA. ACM.

M. Hu and B. Liu. 2004. Mining and summarizing cus-

tomer reviews. SIGKDD '04, pages 168–177, NY, USA. ACM.

M. Hu and B. Liu. 2006. Opinion extraction and summarization on the web. In *The National Conference On Artificial Intelligence*, volume 21, page 1621. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999.

J.S. Justeson and S.M. Katz. 1995. Technical terminology: some linguistic properties and an algorithm for identification in text. *Natural language engineering*, 1(01):9–27.

B Liu, M Hu, and J Cheng. 2005. Opinion observer: analyzing and comparing opinions on the web. WWW '05, pages 342–351, New York, NY, USA. ACM.

O. Medelyan, I.H. Witten, and D. Milne. 2008. Topic indexing with Wikipedia. In *AAAI WikiAI workshop*.

D. Milne and I.H. Witten. 2009. An open-source toolkit for mining Wikipedia. In *Proc. New Zealand Computer Science Research Student Conf., NZCSRSC*, volume 9.

D. Milne. 2007. Computing semantic relatedness using wikipedia link structure. In *New Zealand Computer Science Research Student Conference*. Citeseer.

S. Raju, P. Pingali, and V. Varma. 2009. An unsupervised approach to product attribute extraction. *Advances in Information Retrieval*, pages 796–800.