Opinion Analysis on CAW 2.0 Datasets

Lun-Wei Ku
Department of Computer Science and Information Engineering
National Taiwan University
No. 1, Sec. 4, Roosevelt Road,
Taipei, 10617 Taiwan
+886-2-33664888 ext 301
lwku@nlg.csie.ntu.edu.tw

Kai-Jie Ke
Department of Computer Science and Information Engineering
National Taiwan University
No. 1, Sec. 4, Roosevelt Road,
Taipei, 10617 Taiwan
+886-2-33664888 ext 301
kjko@nlg.csie.ntu.edu.tw

Hsin-Hsi Chen
Department of Computer Science and Information Engineering
National Taiwan University
No. 1, Sec. 4, Roosevelt Road,
Taipei, 10617 Taiwan
+886-2-33664888 ext 311
hhchen@ntu.edu.tw

ABSTRACT

In this paper, we aim to classify non-domain-specific web articles into three types: factual, opinion positive and opinion negative, on the Web 2.0 corpus provided by the CAW 2.0 workshop. We investigate the fields in these Web 2.0 articles and extract features from their contents for developing the classifier. On the one hand, to fully utilize the characteristics of Web 2.0 data, the metadata of the articles are extracted as features for the classifier. On the other hand, words, word sequences, and linguistic cues are also extracted to represent the contents of the articles. An SVM classifier is applied here. Features extracted from the fields available in one dataset are compared with those available in all datasets. Moreover, to extract the sentence structures and grammatical relations as features, a parser is applied on the strings of body contents. Experiment results show that metadata are useful in the classification process, and despite of the evaluating score, the helpfulness field and the author field are two most useful features. As the training data released by CAW 2.0 workshop are categorized into recommended and not recommended, accuracy 0.985, f-score 0.992 for retrieving recommended comments and f-score 0.871 for retrieving not recommended comments are achieved by the proposed SVM classifier using features extracted from both metadata and contents of comments. Then a method for mapping our results to the requested answers of CAW 2.0 workshop is proposed to generate the testing results in the opinion analysis track.

Categories and Subject Descriptors

H.3.3 [**INFORMATION STORAGE AND RETRIEVAL**]: Information Search and Retrieval – *search process, selection process.*

General Terms

Algorithms, Performance, Experimentation, Human Factors, Languages.

Keywords

Information extraction, Opinion analysis, sentiment analysis.

1. INTRODUCTION

Opinion analysis has become a hot research topic, not only because of its applicability but also its close relationship to people and the Web. With the easily accessible and publishable environment on the Web, the number of articles containing opinions is increased enormously. Automatically analyzing opinions is an important approach to capture the feelings and thoughts of the public. Moreover, under the framework of Web 2.0, clean metadata and content data are more available than ever before from various web articles. This facilitates the opinion analysis by skipping the tedious preprocessing work.

Opinion analysis has been widely investigated since 2005. This research topic can be discussed from different aspects such as granularities of information, information sources, and methodologies. Documents, sentences, and words express three different levels of subjective information for opinion mining. Pang, Lee, and Vaithyanathan [1] classified documents by overall sentiments instead of topics. Riloff and Wiebe [2] distinguished subjective sentences from objective ones. Kim and Hovy [3] presented a sentiment classifier for English words and sentences by utilizing thesauri. Later researchers concerned not only the subjectivity, but also the polarity of these subjective elements. The simplest opinion polarity set contains binary values: positive and negative. A more complex model may classify opinion polarities into several levels [4] or even report their strength by scores [5].

Information source of opinions is another issue in this research topic. Reviews are often adopted for opinion analysis due to their practicalities. Dave, Lawrence and Pennock [6] extracted opinions from product reviews. Also for reviews, Liu, Hu and Cheng [7] illustrated opinion summarization of bar graph style. Bai, Padman and Airoldi [8] categorized movie reviews by opinion polarities. Ghose and Ipeirotis [9] further ranked reviews from end users' and manufacturers' aspects using sentiment information to provide useful suggestions. However, adopting reviews as the experimental materials usually implies that the proposed approaches are restricted to services or products.

In this paper, we utilize the training data released by CAW 2.0 workshop. In the sentiment and opinion analysis task of CAW 2.0, the proposed approach has to report whether a comment is factual, opinion positive, or opinion negative, and assign it a degree of association. In this case, the proposed approach has to determine both subjectivity and polarity. As the materials are comments, the units for opinion analysis here are more likely to be documents and sentences. In some extreme cases, they could be words, too. These comments, according to the descriptions from the workshop, are from web sites Twitter, MySpace, Slashdot, Ciao and Kongregate, which contain the daily interactions within a virtual community. Hence we have to analyze opinions in a more general domain compared to earlier research.

According to the descriptions on the official web site of the workshop, the training data should be restricted to the provided

dataset. Due to this restriction, the opinion classification problem is viewed as a machine learning problem in this paper. We build an SVM classifier to report whether the given comments are factual, opinion positive, or opinion negative. Two different types of features were utilized here: un-structural features and structural features. Un-structural features were extracted directly from fields in training comments. Structural features, instead, were sentential grammatical relations extracted from parsed comments. From the experimental results utilizing these two kinds of features, we expect to know whether structural information is useful for web comments while they generally are considered not well written.

We also compared features designed from a specific web site and features commonly seen in most web sites. From this aspect, we expect to know whether using specific metadata provided by one web site will improve the performance.

2. SVM Approach

SVM is a commonly used machine training algorithm. We adopt libSVM [10] to develop our classifier. Training comments are stemmed before we extract features from them.

Each comment provided by CAW 2.0 contains metadata and its body contents. Among five training data sets including Twitter, MySpace, Slashdot, Ciao and Kongregate, Ciao contains the "recommended" field which is approaching the opinion polarity we try to learn. Therefore, we treat comments whose values in the recommended field are true as opinion positive comments, while those whose values are false as opinion negative comments for training.

3. Features

As mentioned, only the Ciao dataset has the field containing information approaching the answer. Therefore, we selected features from fields in this dataset for training. However, we are not certain that whether these fields are also available in other datasets. Therefore, we experiment with a common feature set as well. In this feature set, features are selected from the Ciao fields only when these fields are also available in other datasets. Moreover, we extract grammatical relations as features from the body contents of Ciao comments to experiment about whether the linguistic cues are useful for Web 2.0 data.

3.1 Extracting Features from Ciao Fields

Information in the recommended field is extracted to simulate the correct answer. Except the recommended field, there are 10 other fields in comments of the Ciao dataset. Five fields among them are selected as the sources of our features:

(1) User ID. We want to monitor the user behavior from this feature. There may be some users who tend to give positive comments, while some other users negative comments. The probabilities for each user to post positive comments are calculated and treated as a feature. Notice that, though we believe that this feature is useful, it is not valid when working on the datasets from the other Web 2.0 sites because the virtual communities in different sites are surely different. However, in the same site, even the collecting period of the testing data is different from that of the training data, the previous behaviors of users revealed by the training data are still referable.

(2) Post Score. The value of this field ranges from 0 to 50. This value denotes the helpfulness of each review in making a buying decision. This value represents the evaluation of other readers on the current review by averaging all readers' ratings. This helpfulness information may appear in stars in some web sites.

(3) Title. The titles of the comments are often the evaluated targets in this thread. There may be the possibility that most people have same opinions toward same services or products. Therefore we included the title string as our features. The way we used it is to calculate how many times bigrams and trigrams of title strings appear in the body field and took these two values as two features.

(4) Date. The date of each comment records the time it is posted. It is represented as a timestamp in the Ciao dataset. This value is used as a feature to monitor the burst effect of opinions along the timeline.

(5) Body. The body field contains the content string of each comment. People should be able to judge whether the comment is factual, opinion positive, or opinion negative solely from this string. We extracted unigram and bigram terms of this string, and their frequencies in each comment were calculated and treated as features.

In sum, we have 1,047,442 features extracted. Other fields not utilized in Ciao include average rating of the current product by readers (one for each thread), an introduction of the current product (one for each thread), the positive points given by the comment author (one for each comment) and the negative points given by the comment author (one for each comment). Thread Score, the value denotes the rating of the service or product in this thread, was adopted first in the experiments. It ranges from 0 to 50 by adding 10 each time, and is assigned by the author of this comment. However, we find that it almost perfectly matched with what we are going to predict: recommended or not recommended (see results shown in Table 4), and this feature dominate other features. Therefore, we decide not to use this feature in our experiments.

3.2 Features Extracted from Common Fields

In the previous section, we discuss the way of extracting features from fields in the Ciao dataset. However, the fields in the Ciao dataset do not necessarily appear in the other datasets. For comparison, we extract features only from those fields existing in all datasets to train our classifier. We find that only two fields, User ID and Body, exist in all datasets. Two commonly seen fields, Title and Date, do not exist in dataset Kongregate and Twitter, respectively. Therefore, we also consider these two fields in the other experiment to loose the constraint. In sum, we have two settings to test the performance of using commonly seen fields: extracting features from User ID and Body, and extracting features from User ID, Body, Title, and Date.

3.3 Structural Features

To extract structural features, we utilized the Stanford parser[1] to pre-process all comments. Our purpose is to extract the grammatical relations from dependency trees. Ku *et al.* [11] mentioned that some structural relations in Mandarin are beneficial for opinion analysis, and we find that these relations are similar to some grammatical relations in dependency trees [12].

[1] http://nlp.stanford.edu/software/lex-parser.shtml

Therefore, we select those grammatical relations that may bear opinions according to Ku's research as features. These relations are as follows:

(1) acomp: adjectival complement. Adjectives may bear opinions in this relation.

(2) advmod: adverbial modifier. Adverbs may bear opinions.

(3) amod: adjectival modifier. Adjectives may bear opinions.

(4) cop: copula. The complement of the copular verb may bear opinions.

(5) dobj: direct object. The verb and the accusative object together may bear an opinion.

(6) mark: marker. Markers like *because*, *when*, and *although* could change the tone in sentences and thus may be an indicator for the appearance of opinions.

(7) nn: noun compound modifier. Nouns modifying the head noun may bear opinions.

(8) nsubj: nominal subject. The verb or the complement of the copular verb in this relation may bear opinions.

(9) rcmod: relative clause modifier. The relative clause of the noun phrase may bear opinions.

(10) xcomp: open clausal complement. The open clausal complement is a clausal complement without its own subject. This complement may bear opinions.

Two components form a relation. The frequencies of each relation, the relation and the first component pair, the relation and the second component pair, the relation plus the component pair are calculated from the training dataset and utilized as features.

An important grammatical relation that may inverse the polarity of opinions is also included:

(11) neg: negation modifier. This relation will reverse the polarity of other relations which contains the negated component in this relation. Therefore, if this relation appears, the appearance of other relations which contains the negated component in this relation will be calculated separately. For example, if an advmod relation contains a negated component, its frequency will be counted and added to the neg-advmod relation instead of the advmod relation.

4. Experiment Results and Discussions

We first adopted five features mentioned in Section 3.1 for our experiment. If features except content in the body field are adopted, that is, only metadata are adopted, we obtain accuracy 0.957. We then add the unigrams in the body contents as features and obtain accuracy 0.992. However, we found that we have these surprising results because the experimental materials are seriously unbalanced. In all 20,879 comments in the Ciao dataset, 19,653 comments are annotated as recommended but only 1,226 comments are annotated as not recommended. That is, over 94% comments are annotated as recommended in this dataset. Therefore, the developed SVM classifier annotated almost all comments as recommended.

To avoid this deceiving good performance obtained from the experiments on an unbalanced dataset, we redesign the experimental methodology for all experiments. We ran a four-fold experiment on the Ciao dataset. For each fold, we randomly generated a balanced training set from three fourth of the Ciao dataset to train our classifier, but left the developing set, which is one forth of the Ciao set, unchanged to simulate the distribution of the real data. Table 1, 2 and 3 show the four-fold experimental results on the released training Ciao set. Note that this four-fold experiment methodology is applied throughout this section, and the figure of accuracy, precision, recall and f-score reported is the average of four-fold experiments.

Table 1. Experiments Results (accuracy)

Settings	Accuracy
Features-B	0.951
Features (uni)	0.984
Features (bi)	0.962
Features (uni+bi)	0.985

Table 2. Experiments Results (retrieving recommended posts)

Settings	Precision	Recall	f-score
Features-B	0.999	0.948	0.973
Features (uni)	0.993	0.989	0.991
Features (bi)	0.997	0.962	0.980
Features (uni+bi)	0.992	0.992	0.992

Table 3. Experiments Results (retrieving not recommended posts)

Settings	Precision	Recall	f-score
Features-B	0.543	0.982	0.700
Features (uni)	0.837	0.882	0.859
Features (bi)	0.614	0.960	0.749
Features (uni+bi)	0.872	0.869	0.871

Features-B adopts features extracted from all fields in Section 3.1 except Body field. Features (uni), Features (bi) and Features (uni+bi) included featuresfrom all five fields, but Features (uni) extracted unigrams, while Features (bi) extracted bigrams and Features (uni+bi) extracted both unigrams and bigrams from the Body field. An interesting result is observed here: using features extracted from metadata already achieves a satisfactory result (Features-B). Although people can always classify the comments by its contents, understanding opinion related contents for classification systems is not always easy. Considering metadata may provide another possible approach for opinion analysis.

Table 4. Results of using features extracted from each field (accuracy)

Field	Accuracy
User ID	0.754
Thread Score	1.000
Post Score	0.792
Title	0.245
Date	0.491
Body (uni)	0.708
Body (uni+bi)	0.940

Table 5. Results of using features extracted from each field (retrieving recommended posts)

Field	Precision	Recall	f-score
User ID	0.994	0.746	0.852
Thread Score	1.000	0.999	0.999
Post Score	1.000	0.779	0.876
Title	0.954	0.209	0.343
Date	0.950	0.483	0.640
Body (uni)	0.968	0.720	0.826
Body (uni+bi)	0.942	0.999	0.969

Table 6. Results of using features extracted from each field (retrieving not recommended posts)

Field	Precision	Recall	f-score
User ID	0.185	0.927	0.309
Thread Score	0.982	1.000	0.991
Post Score	0.221	1.000	0.361
Title	0.062	0.840	0.116
Date	0.067	0.593	0.120
Body (uni)	0.122	0.622	0.205
Body (uni+bi)	0.145	0.011	0.021

Results in Table 4, 5, and 6 also reveal some interesting phenomena. First, as mentioned in Section 3.1, the value in Thread Score field can perfectly classify comments into recommended ones and not recommended ones. In other words, Thread Score represents the rating from the author and it is consistent with the author's final decision to recommend or not to recommend. The value in Post Score field is the second useful feature. It represents the helpfulness of each comment rated by other readers, but helpfulness is not one hundred percently consistent with author's recommending decision for services and products. Not recommended comments can be as helpful for readers as recommended ones.

The performance of using User ID as features, surprisingly, is ranked the third. The usefulness of this feature may indicate that some people tend to have a positive attitude, and some people like to make criticisms all the time. However, this feature is not easily available in the testing set. Recall that we utilized User ID by calculating the percentage of recommended comments posted by this ID, and we have to know whether comments in hand are recommended or not to calculate this number. Moreover, we must work on the same virtual community in both the training and the testing phase. Nevertheless, the goal of utilizing User ID in this way is to verify whether it is useful. Since we have confirmed its usefulness, we may still utilize this feature in a different way in the testing phase. For example, we can first classify comments without this feature and obtain the initial classification results. Then we may calculate the percentage of recommended comments posted by each author using those results with high confidence to obtain User ID features. Next we may retrain the classifier together with the User ID features, and get new results from the retrained classifier. This process can be performed iteratively until the results converge.

The results also show that features extracted from the Title field and Date field are not really beneficial. Moreover, if there is no available language resource, using n-grams in the Body field as features will not perform well. Results show that adding bigram features will always decrease the performance. This may be due to the sparseness problem, because we have lots of bigram features but the length of comments is generally shorter than the length of formal articles. Some comments contain even only one sentence or one word. A better way to utilize the content strings is to extract sentiment or subjective words first before calculating the frequency of unigrams and bigrams. Therefore, having resources to help identify subjective information is necessary for this task.

Table 7, 8 and 9 show the results of extracting features from common fields in five datasets, and commonly seen fields. The former, common fields, include only User ID and Body, while the latter, commonly seen fields, include User ID, Body, Title and Date.

Table 7. Results of extracting features from common and commonly seen fields (accuracy)

Setting	Accuracy
Common	0.918
Commonly Seen	0.542

Table 8. Results of extracting features from common and commonly seen fields (retrieving recommended posts)

Setting	Precision	Recall	f-score
Common	0.972	0.942	0.956
Commonly Seen	0.953	0.536	0.686

Table 9. Results of extracting features from common and commonly seen fields (retrieving not recommended posts)

Setting	Precision	Recall	f-score
Common	0.374	0.559	0.448
Commonly Seen	0.071	0.572	0.127

Table 7 shows that using features from common fields are better than using features from commonly seen fields. Features from the Title and Date fields are not beneficial to classification and may deteriorate the performance, as we have seen in Table 4 to 6. However, neither of them is better than using features from all fields. We can conclude that specific information provided by each web site is likely to be useful for extracting opinions. Specific fields may capture the native characteristics of the web site.

Table 10. Results of extracting features from dependency relations (accuracy)

Setting	Accuracy
Relation (all, none)	0.644
Relation (all, com1 or com2)	0.668
Relation (all, com1 and com2)	0.672
Relation (11, none)	0.739
Relation (11, com1 or com2)	0.676
Relation (11, com1 and com2)	0.687

Table 11. Results of extracting features from dependency relations (retrieving recommended posts)

Setting	Precision	Recall	f-score
Relation (all, none)	0.962	0.643	0.771
Relation (all, com1 or com2)	0.983	0.654	0.785
Relation (all, com1 and com2)	0.982	0.659	0.789
Relation (11, none)	0.952	0.760	0.845
Relation (11, com1 or com2)	0.985	0.665	0.794
Relation (11, com1 and com2)	0.985	0.676	0.802

Table 12. Results of extracting features from dependency relations (retrieving not recommended posts)

Setting	Precision	Recall	f-score
Relation (all, none)	0.094	0.598	0.163
Relation (all, com1 or com2)	0.129	0.817	0.222
Relation (all, com1 and com2)	0.129	0.809	0.223
Relation (11, none)	0.103	0.418	0.165
Relation (11, com1 or com2)	0.143	0.845	0.244
Relation (11, com1 and com2)	0.147	0.845	0.250

Table 10, 11 and 12 show the results of adopting dependency relations as features. The settings in these tables vary in two aspects: adopted relations and involved components, and represented as *Relation (adopted relations, involved components).* As mentioned, two components form a relation. For adopted relations, there are two options: considering all relations (all) and considering 11 relations (11) mentioned in Section 3.1 when extracting features. For involved components, there are three options: in the setting "none", no components are considered and only relations themselves are extracted as features; in the setting "com1 or com2", the relation plus the first component, and the relation plus the second component, are both extracted as features; in the setting "com1 and com2", all three terms, relation, the first and the second components, are extracted as one feature for the classifier. Results show that considering the relation, first component, and second component as one feature achieves best results when considering Table 11 and 12 together. That is, only considering relations may not give us enough information whether the content of the comment is recommending products or services. The composite words of relations may provide more semantic information. Moreover, considering only 11 selected relations is better than considering all relations. This shows that feature selection can improve the results in our approach.

Adopting relations as features can achieve better results to using n-grams of the Body field as features. Although parsing costs time, it could be a better choice for non-real-time applications. This result also tells us that linguistic cues are useful in the opinion analysis track. As relations and n-gram features are both extracted from the Body field, they are joined in next experiments. Results are shown in Table 13, and they are worse than only using either kind of features. It is not what we expect to see. The main reason may be the sparseness problem of both kinds of features again, and an appropriate method to incorporating these two kinds of features is required.

Table 13. Results of adopting features of dependency relations and n-grams

Retrieving	Precision	Recall	f-score
Recommended	0.942	0.425	0.585
Not-recommended	0.060	0.584	0.108

5. Mapping Predicted Results to the Required Answer Set

The opinion analysis track of CAW 2.0 workshop requests the developed system to assign a given comment with a degree of association to three basic categories: factual, opinion positive and opinion negative. The degree of association for each category should be provided by means of a real value within the range between zero (not associated at all) to one (completely associated). However, the developed SVM classifier only classifies comments into two categories by the learned separating hyper-plane. Therefore, we modified the libSVM source code to print out the distance of each data point to this separating hyper-plane. Then we can find the most distant data point among the recommended ones and the not recommended ones, P-rc and P-nrc, respectively. We postulate that the data points close to the separating hyper-plane are likely representing the factual comments, while those close to P-rc recommended comments and those close to P-nrc not recommended comments. We calculate the distances from the current data point to the separating hyper-plane, P-rc, and P-nrc, and d-hp, d-rc, and d-nrc indicate these three distances, respectively. The inverse of them, 1/d-hp, 1/d-rc, and 1/d-nrc, are normalized and reported as the degree of association to the factual, opinionated-positive and opinionated-negative categories, respectively.

6. Conclusion and Future Work

The CAW 2.0 opinion analysis task provides clean Web 2.0 data crawled from five sites: Ciao, Kongregate, MySpace, Slashdot, and Twitter. The aim of this task is to automatically classify factual, opinion positive, opinion negative comments and assign a degree of association to these three categories. In the training phase, the experimental materials are restricted to the released datasets. Because the released datasets contain no gold standard, we utilized information in the "recommended" field provided by the Ciao dataset to simulate the correct answers for opinion positive and opinion negative comments in our experiments. We train an SVM classifier for this task by different feature sets: features extracted from fields in Ciao and features extracted from common fields. Experiments show that metadata are beneficial for opinion analysis in the Web 2.0 datasets. Among Ciao's metadata fields, helpfulness information given by readers (Post Score) and, surprisingly, User ID are two most useful feature sources. We also find that only using n-grams features of contents without incorporating language resources is not good enough to provide satisfactory results, and the sparseness problem might be the main issue.

As to the linguistic cues, experiments show that using features extracted from dependency relations performs better than using n-gram features. However, joining them together will not bring better results. A more appropriate way to incorporate these two different types of features is required.

All experiments were done in a four-fold mode. We have achieved accuracy 0.985, f-score 0.992 for retrieving

recommended comments and f-score 0.871 for retrieving not recommended comments on the Ciao dataset. Features extracted from User ID, Post Score, Title, Date, and Body fields trained the best classifier. However, these are results from our simulating experiments: we do not have the gold standard for evaluation in the training phase. We have found that n-grams are not good enough for detecting opinion polarities. We plan to improve our classifier by the following two possible approaches:

(1) Feature selection. The sparseness problem may deteriorate the performance of the classifier. We can filter out low frequency n-grams, or drop some features that are not so helpful, for instances, factual words or words of some specific part of speech, to decrease noise and mitigate the sparseness problem. Also features extracted by calculating collocation information could be considered in the future.

(2) Adopting language resources. Although the experimental materials for the opinion analysis track are restricted to the released datasets, we may still add features according to different language resources to know whether they are useful in the future. For example, we can use dictionaries to obtain prefixes and suffixes of words, utilize sentiment dictionaries like General Inquire [2] and SentiWordnet [3], or train with manually annotated corpus like MPQA[4].

We adopt the distance provided by the libSVM to map our results to the requested answers of CAW 2.0 workshop. Whether the mapping method is appropriate is also an interesting research topic after the evaluation results of the testing data are available.

7. REFERENCES

[1] Pang, B., Lee, L. and Vaithyanathan, S.. 2002. Thumbs up? Sentiment Classification Using Machine Learning Techniques. Proceedings of the 2002 Conference on EMNLP, 79-86.

[2] Riloff, E. and Wiebe, J.. 2003. Learning Extraction Patterns for Subjective Expressions. Proceedings of the 2003 Conference on EMNLP, 105-112.

[3] Kim, S.-M. and Hovy, E.. 2004. Determining the Sentiment of Opinions. Proceedings of the 20th ICCL, 1367-1373.

[4] Wilson, T., Wiebe, J. and Hoffmann, P., Recognizing Contextual Polarity in Phrase-Level Sentiment Analysis, Proceedings of HLT/EMNLP 2005, 2005, 347-354.

[5] Ku, L.-W. and Chen, H.-H.. 2007. Mining Opinions from the Web: Beyond Relevance Retrieval. Journal of American Society for Information Science and Technology, Special Issue on Mining Web Resources for Enhancing Information Retrieval, 58(12), 1838-1850.

[6] Dave, K., Lawrence, S., and Pennock, D.M.. 2003. Mining the Peanut Gallery: Opinion Extraction and Semantic Classification of Product Reviews. Proceedings of the 12th International World Wide Web Conference, 519-528.

[7] Liu, B., Hu, M. and Cheng, J. (2005). Opinion Observer: Analyzing and comparing opinions on the web. Proceedings of the 14th International World Wide Web Conference, 342-351.

[8] Bai, X., Padman, R. and Airoldi, E. (2005). On learning parsimonious models for extracting consumer opinions. Proceedings of the 38th Annual Hawaii International Conference on System Sciences, Track 3, Volume 03, 75.2.

[9] Ghose, A. and Ipeirotis, P. (2007). Designing novel review ranking systems: Predicting usefulness and impact of reviews. Proceedings of the International Conference on Ectronic Commerce (ICEC), Invited paper, 303-310.

[10] Chang, Chih-Chung and Lin, Chih-Jen. 2001. LIBSVM: a library for support vector machines, http://www.csie.ntu.edu.tw/~cjlin/libsvm

[11] Ku, L.-W., Liu, I.-C., Lee, C.-Y., Chen, K.-h. and Chen, H.-H. (2008). Sentence-Level Opinion Analysis by CopeOpi in NTCIR-7. Proceedings of the 7th NTCIR Workshop Meeting on Evaluation of Information Access Technologies: Information Retrieval, Question Answering, and Cross-Lingual Information Access, December 16-19, 2008, Tokyo, Japan, 260-267.

[12] Marie-Catherine de Marneffe, Bill MacCartney and Christopher D. Manning. 2006. Generating Typed Dependency Parses from Phrase Structure Parses. Proceedings of the5th international conference on Language Resources and Evaluation (LREC 2006), 449-454

[2] http://www.wjh.harvard.edu/~inquirer/

[3] http://sentiwordnet.isti.cnr.it/

[4] http://www.cs.pitt.edu/mpqa/