Feature Specific Sentiment Analysis for Product Reviews

Subhabrata Mukherjee, Pushpak Bhattacharyya
Dept. of Computer Science and Engineering, IIT Bombay
{subhabratam, pb}@cse.iitb.ac.in

Abstract: In this paper, we present a novel approach to identify *feature specific* expressions of opinion in product reviews with *different features* and *mixed emotions*. The objective is realized by identifying a set of potential features in the review and extracting opinion expressions about those features by exploiting their associations. Capitalizing on the view that more closely associated words come together to express an opinion about a certain feature, dependency parsing is used to identify relations between the opinion expressions. The system learns the *set of significant relations* to be used by dependency parsing and a *threshold parameter* which allows us to merge closely associated opinion expressions. The data requirement is minimal as this is a *one time learning* of the *domain independent parameters*. The associations are represented in the form of a graph which is partitioned to finally retrieve the opinion expression describing the *user specified feature*. We show that the system achieves a *high accuracy across all domains* and performs at par with state-of-the-art systems despite its data limitations.

1 Introduction

In recent years, the explosion of social networking sites, blogs and review sites provide a lot of information. Millions of people express uninhibited opinions about various product features and their nuances. This forms an active feedback which is of importance not only to the companies developing the products, but also to their rivals and several other potential customers.

Sentiment Analysis is the task of tapping this goldmine of information. It retrieves opinions about certain products or features and classifies them as *recommended* or *not recommended*, that is *positive* or *negative*.

The sentiment regarding a particular product in a review is seldom explicitly positive or negative; rather people tend to have a mixed opinion about various features, some positive and some negative. Thus the feature specific opinion matters more than the overall opinion.

Consider a review *"I like Micromax's multimedia features but the battery life sucks."* This sentence has a mixed emotion. The emotion regarding *multimedia* is positive whereas that regarding *battery life* is negative. Hence, it is of utmost importance to extract only those opinions relevant to a particular feature (like *battery life or multimedia*) and classify them, instead of taking the complete sentence and the overall sentiment.

In this work, we propose a method that represents the features and corresponding opinions in the form of a graph where we use dependency parsing to capture the relations between the features and their associated opinions. The idea is to capture the

association between any specific feature and the expressions of opinion that come together to describe that feature. This is done by capturing the *short range* and *long range dependencies* between the words using dependency parsing. Clustering is done on the graph to retrieve only those opinion expressions that are *most closely related* to the *target feature* (user specified feature) and the rest are pruned. We apply merging in the final phase of our algorithm to merge the opinions about any 2 features that cannot be described independent of each other. We apply our method to domain specific reviews to test the efficacy of the system. We achieved a high accuracy across all domains over the baseline. We compare our approach with state-of-the-art systems [4] where we achieve a comparable accuracy despite data limitations. The system performance improved greatly not only over the naïve baseline but also over the chosen improved baseline [5].

The roadmap to the remaining part of the paper is as follows:
Section 1 presents the motivation and objective of the current work. Section 2 gives a related work section. Section 3 defines the problem statement. Section 4 gives the algorithm to extract features and their associated opinion expressions. It presents a graph based representation of the features and their relations, which is partitioned to obtain feature specific opinions. A rule-based and supervised classification system is presented in Section 5 to find the final sentiment polarity. We present the learning of the domain independent parameters in Section 6, followed by extensive experiments across various product domains in review blogs to validate our claim. Section 7 gives the conclusions and directions for future work followed by references.

2 Related Work

Chen *et. al* [1] use dependency parsing and shallow semantic analysis for Chinese opinion related expression extraction. They categorize relations as, topic and sentiment located in the same sub-sentence and quite close to each other (like the rule "an adjective plus a noun" is mostly a potential opinion-element relation), topic and sentiment located in adjacent sub-sentences and the two sub-sentences are parallel in structure (that is to say, the two adjacent sub-sentences are connected by some coherent word, like although/but, and etc), topic and sentiment located in different sub-sentences, either being adjacent or not, but the different sub sentences are independent of each other, no parallel structures any more.

Wu *et. al* [2] use phrase dependency parsing for opinion mining. In dependency grammar, structure is determined by the relation between a head and its dependents. The dependent is a modifier or complement and the head plays a more important role in determining the behaviors of the pair. The authors want to compromise between the information loss of the word level dependency in dependency parsing as it does not explicitly provide local structures and syntactic categories of phrases and the information gain in extracting long distance relations. Hence they extend the dependency tree node with phrases."

Hu *et. al* [3] used frequent item sets to extract the most relevant features from a domain and pruned it to obtain a subset of features. They extract the nearby adjectives to a feature as an *opinion word* regarding that feature. Using a seed set of labeled Adjectives, which they manually develop for each domain, they further expand it using WordNet and use them to classify the extracted opinion words as positive or negative.

Lakkaraju *et. al* [4] propose a joint sentiment topic model to probabilistically model the set of features and sentiment topics using HMM-LDA. It is an unsupervised system which models the distribution of features and opinions in a review and is thus a generative model.

Most of the works mentioned above require labeled datasets for training their models for each of the domains. If there is a new domain about which no prior information is available or if there are mixed reviews from multiple domains inter-mixed (as in *Twitter*), where the domain for any specific product cannot be identified, then it would be difficult to train the models. The works do not exploit the fact that majority of the reviews have a lot of domain independent components. If those domain independent parameters are used to capture the associations between features and their associated opinion expressions, the models would capture majority of the feature specific sentiments with minimal data requirement.

3 Problem Statement

Given a product review containing multiple features and varied opinions, the objective is to extract expressions of opinion describing a *target feature* and classify it as positive or negative. The objectives can be summarized is:

1. Extract all the features from the given review

   In the absence of any prior information about the domain of the review (in the form of untagged or tagged data belonging to that domain), this will give a list of *potential features* in that review which needs to be pruned to obtain the exact features.

   Consider the review, *"I wonder how can any people like Max, given its pathetic battery life, even though its multimedia features are not that bad."*

   Here, *multimedia features* and *battery life* are the exact features pertaining to the *mobile domain*. But without any prior domain information, we can use an approximate method to obtain a list of potential features that may include other noisy features as well, example *people*. So this list needs to be pruned to remove the noise and obtain the exact set of features.
2. Extract opinion words referring to the target feature

   The opinion words are not only Adjectives like *hate, love* but also consist of other POS categories like Nouns (*terrorism),* Verbs (*terrify)* and Adverbs (*gratefully*). A naïve method, like extracting the opinion words closest to the *target feature*, does not work so well when the sentence has multiple features and distributed emotions (as we will see later).

   In the example above, *pathetic* and *not bad* are the opinion expressions referring to *battery life* and *multimedia features* respectively.
3. Classify the extracted opinion words as positive or negative

   This step will mark *pathetic* as a negative opinion and *not bad* as a positive opinion.

4 Feature Specific Sentiment Analysis

In this section, we will first outline a method to extract features and their associated relations.

4.1 Feature Extraction

We will elaborate 2 methods for extracting features corresponding to the availability of domain knowledge.

4.1.1 Feature Extraction in Absence of Domain Knowledge

In the absence of any prior information about the product domain, we can make a list of *potential features* in the review by constraining the features only to be Nouns (Example: *multimedia, firmware, display, color* etc.). All the words in the sentence are POS-tagged and all the Nouns are retrieved. Initially, all the Nouns are treated as features and added to the *feature list* ***F***.

Consider the review,
"I have an ipod and it is a great buy but I'm probably the only person that dislikes the iTunes software."

F = {*ipod, buy, person, software*}
This forms our initial feature set. But the intended features are *ipod* and *software* as they are the features specific to the *mobile domain*. We will later present an algorithm to prune this initial feature set, such that any 2 features *strongly related* will be merged. Thus, *buy* will be merged with *ipod* when the target feature is *ipod,* and {*person, software*} will be pruned. If the target feature is *software*, *person* will be merged with *software,* and {*ipod, buy*} will be pruned.

4.1.2 Feature Extraction in Presence of Domain Knowledge

If domain information is available (in the form of crawled reviews from the domain in focus, when the product domain has been identified) we can extract all the features in the domain using *Latent Dirichlet Allocation* (Blei *et. al* [6]) or *HMM-LDA* (Griffiths *et. al* [7]). In presence of domain knowledge, we would readily know that *software* and *ipod* are mobile-domain specific features whereas *buy* and *person* are not. Using this information we can directly prune the feature list ***F***.

4.2 Relation Extraction

Relation extraction is necessary to identify the associations between the opinion expressions in a review. We will shortly formulate our hypothesis that necessitates this phase. We identify two kinds of relations between the words in a sentence that associate them to form a coherent review:

1. Direct Neighbor Relation

Let *Stopwords* be the list of pre-compiled stop words occurring frequently in any text. This comprises mainly of *be* verbs, personal pronouns, prepositions, conjunctions etc. All Nouns, Adjectives, Adverbs, Verbs (except *be* verbs) are excluded from the list.

Consider a sentence *S* and 2 consecutive words $w_i, w_{i+1} \in S$. *If* $w_i, w_{i+1} \notin Stopwords$, then they are directly related. This helps us to capture *short range dependencies*.

2. Dependency Relation

Let *Dependency_Relation* be the list of significant relations. We call any *dependency relation* significant, if

- It involves any *subject, object* or *agent* like *nsubj, dobj, agent* etc
- It involves any *modifier* like *advmod, amod* etc
- It involves *negation* like *neg*
- It involves any *preposition* like *prep_of*
- It involves any *adjectival or clausal* component like *acomp, xcomp*

The above set of relations is not minimal, in the sense that not all of them are equally significant in capturing the semantic coherence in reviews. We will later show how to prune the above set of relations, to obtain a minimal set of significant relations, by a small seed set of data using ablation test.
Any 2 words w_i and w_j in *S* are directly related, *if*
$\exists D_l \;\; s.t. \;\; D_l(w_i, w_j) \in Dependency\_Relation$.
This helps us to capture *long range dependencies*.

The direct neighbor and dependency relations are combined to form the master *relation set* ***R***.
We now formulate the following hypothesis:

More closely related words come together to express an opinion about a feature.

If there are *'n'* features of a product in a sentence, then those words that are most closely related (in terms of relations defined above) to a feature *'i'* will come together to express some opinion about it, rather than about some other feature *'j'*, to which they are not so closely associated.

For Example: *"I want to use Samsung which is a great product but am not so sure about using Nokia"*.
Here {*great, product*} are related by an adjectivial modifier relation, and {*product, Samsung*} are related by a relative clause modifier relation. Thus {*great, Samsung*} are transitively related. **Here {*great, product*} are more closely related to Samsung than they are to Nokia.** Thus {*great, product*} come together to express an opinion about the entity "Samsung" than about the entity "Nokia". The adjectivial relation is important as it associates the opinion *great* with *product* and the relative clause modifier relation is significant as it associates *product* with *Samsung*.

These relations are provided by the Dependency Parser. We used the Stanford Dependency Parser (http://nlp.stanford.edu:8080/parser/index.jsp).

4.3 Graph Representation

Given a sentence *S*, let ***W*** be the set of all words in the sentence *S*.

A Graph $G(W,E)$ is constructed such that any $w_i, w_j \in W$ are directly connected by $e_k \in E$, *if* $\exists R_l$ *s.t.* $R_l(w_i, w_j) \in R$.
In other words, in the graph *G* all the words in the given sentence are considered as vertices. Any 2 vertices are connected, if there is any relation between them governed by the relation set ***R***.

4.4 Dependency Extraction

We have the set of all features *F* and a graph *G*. Let $f_t \in F$ be the target feature. For example in Section 4.1, *ipod* or *software* can be the *target* feature i.e. the feature with respect to which we want to evaluate the sentiment of the sentence.

Let there be '*n*' features where *n* is the dimension of *F*. The algorithm for extracting the set of words $w_i \in S$, that express any opinion about the target feature f_t proceeds as follows:

i. Initialize *n* clusters $C_i \forall i = 1..n$
ii. Make each $f_i \in F$ the clusterhead of C_i. The target feature f_t is the clusterhead of C_t. Initially, each cluster consists only of the clusterhead.
iii. Assign each word $w_j \in S$ to cluster C_k *s.t.*
$k = \arg\min_{i \in n} dist(w_j, f_i)$,
Where $dist(w_j, f_i)$ *gives the number of edges, in the shortest path, connecting* w_j *and* f_i *in G.*
iv. Merge any cluster C_i with C_t *if* $dist(f_i, f_t) < \theta$, Where θ is some threshold distance.
v. Finally the set of words $w_i \in C_t$ gives the opinion expression regarding the target feature f_t.

Algorithm 1: Dependency Parsing Based Clustering for Sentiment Analysis

In words, we initialize '*n*' clusters C_i, corresponding to each feature $f_i \in F$ *s.t.* f_i is the clusterhead of C_i. We assign each word $w_i \in S$ to the cluster whose clusterhead is closest to it. The distance is measured in terms of the number of edges in the shortest path, connecting any word and a clusterhead. Any 2 clusters are merged if the distance between their clusterheads is less than some threshold. Finally, the set of words in the cluster C_t, corresponding to the target feature f_t gives the opinion about f_t.

Reviews often have opinions about any specific feature that is closely tied with their opinions about some other feature. Consider the review "*I like Nokia a bit more than Samsung*". Here, the opinion regarding Nokia is positive but that regarding Samsung is *not negative*. Thus, if we evaluate the polarity of this sentence with respect to Samsung, the opinion about Nokia has to be factored in i.e. they are not independent. This is the reason for merging the opinion expressions of 2 features if they are closely associated.

4.5 Feature Specific Opinion Extraction with Example

Consider the following review given in Section 4.1.1, *"I have an ipod and it is a great buy but I'm probably the only person that dislikes the itunes software"*.

As shown there, $\boldsymbol{F}$={*ipod,buy, person and software*}forms our initial feature set (represented by rectangles in figure 3). The target feature is $\boldsymbol{f_t}$ = *ipod*. The graph consists of all the words in the sentence as vertices. All the words are connected by relations defined by the master relation set $\boldsymbol{R}$ (shown by thin edges in figure 3). The target cluster $\boldsymbol{C_t}$ has the clusterhead $\boldsymbol{f_t}$. {***I, have, it***} are closest to *ipod* and are assigned to the corresponding cluster whereas {*great, probably, but, im*} are closest to *buy* and assigned to its corresponding cluster (the assignment is shown by bold arrows in figure 3). Now, *ipod* and *buy* are related through *it*. The intercluster-distance between them is 2 which is less than θ=3 and thus the 2 clusters are merged. So, *buy* with all its members is assigned to the target cluster $\boldsymbol{C_t}$. {*an, is, a*}are ignored as StopWords.

Finally $\boldsymbol{C_t}$ comprises of {*I, have, ipod, it, great,buy, probably, but, im*} which represents the opinion expression about the target feature $\boldsymbol{f_t}$= ipod.

5 Classification of Extracted Features

Now, have the set of opinion words $w_i \in C_t$, that describes the target feature f_t.

Rule Based Classification

We use a sentiment lexicon to find the polarity of each word $w_i \in C_t$. If the number of words tagged positive is greater than that tagged negative, we conclude the sentiment regarding the target feature f_t, to be positive or else negative.

Supervised Classification

Each sentence in the review is represented as a vector consisting of the target feature f_t and its associated opinion words $w_i \in C_t$. These set of vectors are fed into any supervised classification system like the SVM.

6. Learning Parameters

We have two principal parameters to learn, the *significant relation set* and the *merging threshold*.

a. Significant Relation Set

Dependency Parsing gives more than 40 relations, not all of which are equally significant. In order to obtain the subset of relations, which are most significant, we have to probe the entire relation space of $O(2^{40})$ if we use an exhaustive search, which is infeasible. So, we use an alternative approach to find the most significant relations to suit our purpose. We partition the relation space in 3 parts:

- Relations that *should be included* in *R*

These consist of the relations *nsubj, nsubjpass, dobj, amod, advmod, nn, neg.*

- Relations that *should not be included* in *R*

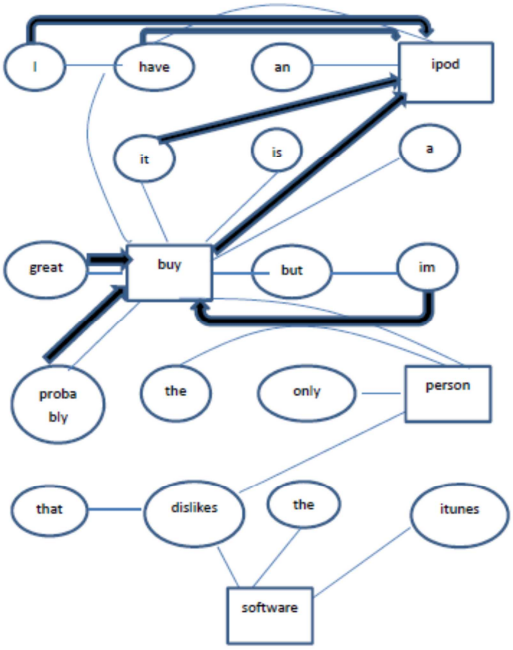

Figure 3: Dependency parsing based Clustering of Features

These consist of relations irrelevant to our purpose like *numeric modifiers*, *abbreviation relations* etc.

- Relations that *may be included* in *R*

This partition consists of around 21 relations which may or may not be significant.

We now perform leave-one-relation out test or *ablation test*. In this, we leave out one relation at a time and compute the overall accuracy of sentiment classification with the remaining relations. Our objective is to find the relations in the 3rd partition that causes *significant* accuracy change. We select an arbitrary domain to perform this test and cross-validate in another domain. We used the labeled data from Hu and Liu *et. al* [5] for learning the parameters.

Table 1: Ablation Test for Significant Relations

Relations	Accuracy (%)
All	63.5
Dep	**67.3**
Rcmod	**65.4**
xcomp, conj_and ccomp, iobj	61.5
advcl , appos, csubj, abbrev, infmod, npavmod, rel, acomp, agent, csubjpass, partmod, pobj, purpcl, xsubj	63.5

In *Table 1*, we find that leaving out *Dep* and *Rcmod* causes significant accuracy improvement, over including all the relations. But, we still cannot be sure which

among *Dep* and *Rcmod* plays the spoilsport. So we perform another experiment in a different domain involving only these 2 relations.

Table 2: Ablation Test for Dep and Rcmod

Relation Set	Accuracy
With Dep+Rcmod	66
Without Dep	**69**
Without Rcmod	67
Without Dep+Rcmod	68

In *Table 2*, we find that Dep causes the real problem. This is also intuitive when we see the definition of the *Dep* relation in Stanford Dependencies Manual which says "*dependency is labeled as dep when the system is unable to determine a more precise dependency relation between two words"*. Thus it captures many stray relations and introduces noise in the graph. Finally, all the relations in Table 1 (excluding *Dep*) are considered as significant relations.

b. Merging Threshold

Any 2 feature clusters are merged if the inter-cluster distance is less than some threshold distance θ. The distance is measured as the number of edges in the shortest path connecting the 2 cluster-heads. If θ is very small, then any 2 clusters having some long-range dependency will not be merged. Whereas if θ is very large, then all the features will be merged and feature specific dependencies will be lost. We used a small seed set from an arbitrary domain to find the optimal value of θ and cross-validated it across other domains.

Table 3: Inter-cluster distance threshold accuracy

θ	Accuracy (%)
2	67.85
3	**69.28**
4	68.21
5	67.40

Table 3 indicates that $\theta = 3$ will give the optimal result. $\theta = 2$ means all the clusters are disjoint and there is no merging, whereas $\theta = 3$ implies any 2 clusters are merged if there is only one intermediate word linking them.

7. Experimental Evaluation

We used 2 datasets. Dataset_1 consisted of 500 reviews extracted from the dataset used by Lakkaraju *et. al* [4]. The extracted data came from 3 domains *laptops, camera and printers*.

The second dataset was extracted from the data used by Hu and Liu *et. al* [5]. It consisted of about 2500 reviews from varied domains like *antivirus, camera, dvd, ipod, music player, router, mobile* etc. Each sentence is tagged with a feature and sentiment orientation of the sentence with respect to the feature.

In the original dataset (*Hu and Liu*, [5]), majority of the sentences consisted of a single feature, and had either entirely positive or entirely negative orientation. From there a new dataset was constructed, by combining each positive sentiment sentence with a negative sentiment sentence using connectives (like *but, however, although), in the same domain, describing the same entity.* For Example, "The display of the camera is bad" and "It is expensive" were connected by *but*. This forms our $Dataset_2$.

Domain	Baseline 1 (%)	Baseline 2 (%)	Proposed System (%)
Antivirus	50.00	56.82	**63.63**
Camera 1	50.00	61.67	**78.33**
Camera 2	50.00	61.76	**70.58**
Camera 3	51.67	53.33	**60.00**
Camera 4	52.38	57.14	**78.57**
Diaper	50.00	63.63	**57.57**
DVD	52.21	63.23	**66.18**
IPOD	50.00	57.69	**67.30**
Mobile 1	51.16	61.63	**66.28**
Mobile 2	50.81	65.32	**70.96**
Music Player 1	50.30	57.62	**64.37**
Music Player 2	50.00	60.60	**67.02**
Router 1	50.00	58.33	**61.67**
Router 2	50.00	59.72	**70.83**

Table 4: Domain specific accuracy for our rule based system in $dataset_2$

Now, each sentence in this new dataset has a mixed emotion about various features.

We determined $Baseline_1$ by counting the number of positive and negative opinion words in the sentence. The final polarity is determined by majority voting. This is a very naïve baseline. So we defined an improved $Baseline_2$ (*Hu and Liu et. al*, [5]). If there '*n*' features f_i and '*m*' opinion words O_i, each O_i expresses an opinion about the nearest feature f_i.

We used the sentiment lexicon used by Hu and Liu *et. al* [5] for rule based classification. Since we have a 2-class classification (positive or negative), any tie is resolved by flipping a coin.

Table 4 gives the domain specific accuracy comparison of our system with $Baseline_1$ and $Baseline_2$. We find that the proposed system performs way better than both the baselines in *every domain*. *Table 5* gives the average accuracy of the system and the baselines across all the domains.

Table 5: Overall accuracy for our rule-based system in $Dataset_2$

System	Accuracy (%)
$Baseline_1$	50.35
$Baseline_2$	58.93
Proposed System	**70.00**

We also performed comparisons with another state-of-the-art system namely, CFACTS developed by Lakkaraju *et. al* [4]. Unlike the CFACTS system, our system

has a much less data requirement as it does not train on domain-specific data. Hence the domain-specific feature extraction accuracy of CFACTS is better. Thus we compared only the final sentiment evaluation accuracy of the 2 systems. This is a valid comparison as CFACTS claimed to have 100% topic purity in feature extraction which means its feature extraction accuracy cannot degrade its sentiment evaluation accuracy.

The performance comparison between the feature specific module of CFACTS and our system is made under the *assumption that the features should be explicitly present in the review*. This is necessary in our system as the user is providing the feature with respect to which the review has to be analyzed. Consider the review sentence, "*The mobile is too heavy*". Here the implicit feature is *weight* and the implicit sentiment is *negative*. Since the system, we developed, does not use any domain specific data for sentiment classification, such reviews cannot be aptly handled by the system.

From *Table 6*, we find that the proposed system performs at par with all the given systems, *with no data requirement*. This, however, comes at a cost that it cannot capture domain-specific implicit feature or hidden sentiment.

In order to have a flavor of the system performance, when tagged data is available, we performed experimental evaluations in 2 arbitrary domains namely, *camera and mobile* using $Dataset_1$.

Table 6: Sentiment Classification accuracy comparison for rule-based classification in $Dataset_1$

System	Sentiment Evaluation Accuracy (%)
$Baseline_1$	68.75
$Baseline_2$	61.10
CFACTS-R	80.54
CFACTS	**81.28**
FACTS-R	72.25
FACTS	75.72
JST	76.18
Proposed System	**80.98**

Table 7: Supervised classification accuracy in 2 domains in $Dataset_2$

Domain	$Baseline_1$ (%)	Proposed System (%)
Mobile	51.42 (50.72/99.29)	**83.82 (83.82/83.82)**
Camera	50	**86.99 (84.73/90.24)**

The supervised system uses Support Vector Machines for classification of feature vectors. *Table 7* shows the huge leap in accuracy from the naïve baseline. The difference in accuracy between the rule-based system and the supervised classification system stems from the fact, that the system can now capture both domain specific sentiment and implicit features. But this comes at a cost of enhanced tagged data requirement for every domain and the system needs to be trained separately for every domain.

8. Conclusions and Future Work

In this paper, we developed a system that extracts potential features from a review and clusters opinion expressions describing each of the features. It finally retrieves the opinion expression describing the user specified feature. The main achievements of the paper can be summarized as:

1. The work exploits associations between the opinion expressions about various features that form a coherent review using dependency parsing.
2. We perform an in-depth analysis of the *dependency relations* deemed significant while mining the relations between the words forming opinion expressions.
3. The system takes into consideration the phenomena where opinion expressions about various features are co-related and thus merges them.
4. The parameters, namely the *significant relation set* and the *merging threshold*, are domain independent. Thus the system has a minimal data requirement as it performs a one-time learning of these parameters.
5. Extensive evaluations were made across various domains over two datasets where the system outperformed the chosen baselines in *all domains*.
6. The system showed improved accuracy not only over the naïve baseline but also over the chosen sophisticated baseline [5].
7. It performed at par with the state-of-the-art systems [4] despite its data limitations, as it does not use any domain specific data for training.
8. We showed that using supervised classification (when tagged data is available for training) the system outperforms the naïve baseline by a huge margin.

The drawback of the system is that it cannot evaluate domain dependent implicit sentiment as it does not train on any domain specific data. Thus the system does not distinguish between “The story is unpredictable” (positive sentiment) and “The steering wheel is unpredictable” (negative sentiment). This is due to the usage of a generic sentiment lexicon, in the final stage, in rule based classification. Supervised classification can distinguish between the two sentiments but it needs tagged data and separate training for every domain.

References

1. Chen Mosha,”Combining Dependency Parsing with Shallow Semantic Analysis for Chinese Opinion-Element Relation Identification”, IEEE, 2010, pp.299-305.
2. Yuanbin Wu, Qi Zhang, Xuanjing Huang, Lide Wu,”Phrase Dependency Parsing for Opinion Mining”, EMNLP '09 Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing, 2009, Volume 3
3. Qi Zhang, Yuanbin Wu, Tao Li, Mitsunori Ogihara, Joseph Johnson, Xuanjing Huang,”Mining Product Reviews Based on Shallow Dependency Parsing”, SIGIR '09, Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval, 2009
4. Himabindu Lakkaraju, Chiranjib Bhattacharyya, Indrajit Bhattacharya and Srujana Merugu,”Exploiting Coherence for the simultaneous discovery of latent facets and associated sentiments”, SIAM International Conference on Data Mining (SDM), April 2011
5. Minqing Hu and Bing Liu, “Mining and summarizing customer reviews”, KDD '04: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining, 2004
6. T. L. Griffiths, M. Steyvers, D. M. Blei and J. B. Tenenbaum,” Integrating Topics and Syntax”, Advances in Neural Information Processing Systems 17, 2005.
7. D. M. Blei, M. Jordan and A. Ng, “Latent Dirichlet Allocation”, Journal of Machine Learning and Research, page 993-1022, 2003.