# Opinion Mining From Noisy Text Data

Lipika Dey and SK Mirajul Haque
TCS Innovation Lab Delhi
Phase 4, Udyog Vihar
Gurgaon, India
{lipika.dey, skm.haque}@tcs.com

## ABSTRACT

The proliferation of Internet has not only generated huge volumes of unstructured information in the form of web documents, but a large amount of text is also generated in the form of emails, blogs, and feedbacks etc. The data generated from online communication acts as potential gold mines for discovering knowledge. Text analytics has matured and is being successfully employed to mine important information from unstructured text documents. Most of these techniques use Natural Language Processing techniques which assume that the underlying text is clean and correct. Statistical techniques, though not as accurate as linguistic mechanisms, are also employed for the purpose to overcome the dependence on clean text. The chief bottleneck for designing statistical mechanisms is however its dependence on appropriately annotated training data. None of these methodologies are suitable for mining information from online communication text data due to the fact that they are often noisy. These texts are informally written. They suffer from spelling mistakes, grammatical errors, improper punctuation and irrational capitalization. This paper focuses on opinion extraction from noisy text data. It is aimed at extracting and consolidating opinions of customers from blogs and feedbacks, at multiple levels of granularity. Ours is a hybrid approach, in which we initially employ a semi-supervised method to learn domain knowledge from a training repository which contains both noisy and clean text. Thereafter we employ localized linguistic techniques to extract opinion expressions from noisy text. We have developed a system based on this approach, which provides the user with a platform to analyze opinion expressions extracted from a repository.

## Categories and Subject Descriptors

H.3.3 [**Informational Storage and Retrieval**]: Information Search and Retrieval – *Information filtering.* I.2.7 [**Artificial Intelligence**]: Natural Language Processing – *text analysis.* I.5.4 **[Pattern Recognition]:** Applications – *text processing*

## General Terms

Algorithms, Experimentation

## Keywords

Noisy text, Opinion mining, Wordnet, Customer feedbacks

## 1. INTRODUCTION

Noisy unstructured text data is generated in informal settings such as online chat, emails, blogs, customer feedbacks and reviews. These texts have the potential to act as rich sources for raw inputs to market research and knowledge discovery. Since Internet is a crucial driving force in today's world, these texts are rich pointers to the collective opinion of the global population on almost every topic. Given the volume and growth rate of these sources, efficient mechanisms are required to aggregate, assimilate and interpret all the information with minimal human intervention. However, the quality of texts generated from these sources can be extremely poor and noisy. Noisy text data typically comprises spelling errors, ad-hoc abbreviations and improper casing, incorrect punctuation and malformed sentences. Hence text mining techniques based on pure linguistic strategies fail to extract information from noisy text. Statistical techniques on the other hand which though not as successful as the linguistic methods, are more suited to extract information from noisy text. However lack of appropriate training data often poses as a bottleneck.

In this paper, we have proposed a hybrid approach to mine opinions from noisy text data comprising blogs, on-line reviews, customer feedbacks on products etc. Opinion mining from noisy text comprises the following sub-tasks:

(i) Pre-processing of noisy text to enable opinion mining. Section 3.1 describes the pre-processing steps. It may be noted that the focus of the system is not to correct all the errors in text. Rather these methods have been designed to focus on minimizing error in opinion mining.
(ii) Locate opinion expressions in processed text – opinion expressions may contain opinionated words and/or opinionated features. We have designed a semi-supervised learning mechanism to identify opinion expressions.
(iii) Analyze opinion expressions to identify features and opinions expressed about them. Opinion is further classified as member of a pre-defined class. The classification is application specific. One can employ a binary classifier to categorize opinions according to their orientations towards positive or negative categories. It is also possible to classify opinions into multiple finer categories like "happiness, sadness, anger" etc. Our system can be trained to operate in either mode. This module uses linguistic rules but within a localized context, to reduce the effects of noisy text.
(iv) Aggregate opinions at multiple levels of granularity as desired by users. Thus opinions could be aggregated feature-wise, product-wise, user-wise, site-wise or at any other level of specificity as desired by users.
(v) The system is accompanied by a Graphical User Interface to facilitate user interaction.

## 2. RELATED RESEARCH

Interest in noisy text analytics has increased significantly in the recent past, and a wide array of methods has been proposed both for noisy text cleaning and analytics. One of the most notable systems in this area was presented in [2], which analyzes Automatic Speech Recognition (ASR) data. ASR data is quite different in nature from blog text. The key challenge there was to detect sentence boundaries since punctuation symbols were missing. Identifying phonetic variants of words was also another challenge for this data.

Blog texts on the other hand suffer from ill-composed sentences, arbitrary punctuations or insertions of characters, irrational capitalization etc. Spelling correction, abbreviation and case restoration mechanisms for noisy text have been addressed in [3, 4]. We have employed variations of these techniques for our work.

Opinion mining is becoming very popular research area in text mining community for its wide applicability. Pioneering work in the area of opinion extraction from customer reviews can be found in [5, 6]. These works employed keyword spotting techniques in collaboration with statistical analysis. WordNet usage for identifying opinion words was proposed in [1]. SentiWordnet [7] is an extension of WordNet, where every sense of every word is assigned a membership to categories positive, negative and neutral. Use of Point-wise Mutual Information (PMI) in [8] has shown the improvement in precision significantly. This work considers some prior knowledge about product properties and parts properties recursively.

Our methodologies are different from all of those stated above. While we also employ key-word spotting to identify opinion words, the key-word collection itself is dynamically generated using WordNet and a semi-supervised learning method. These key-words are then assigned memberships to various opinion classes, based on their observed occurrences in text. However, unlike SentiWordnet, our approach is to attach sentiment to words independent of its sense, since identifying sense of a word is not easy, particularly in noisy text. This method removes the overhead of identifying the sense of the word each time it is encountered in new text to know the sentiment attached to it.

## 3. OPINION MINING PROCESS

In this section, we will present the detailed functioning of the proposed Opinion Mining System. The underlying techniques are fairly robust and achieve high accuracy over both noisy texts and also editorials, reviews and other opinionated clean texts. However, in this paper, we restrict the scope of the presentation to noisy text only. Opinion mining from noisy text progresses through the following steps:

i. A set of pre-defined web sites are crawled and each individual item is stored as a unique document

ii. Noisy text is pre-processed to identify sentence boundaries, correct improper casing, and correct spelling errors with effective utilization of domain knowledge. Case correction is important for the Natural language tools to work properly. Correcting spelling mistakes in a domain dependent fashion is another very important pre-processing task since identification of opinion words and opinionated product features is dependent on this.

iii. The corrected text is subjected to Part Of Speech (POS) and dependency analysis. The phrase structure tree and the dependency relations are used to analyze the opinions.

iv. Use domain knowledge to aggregate opinions at multiple levels of specificity to allow the user to slice and dice information embedded inside text at desired levels.

The remaining sub-sections present the functional details of the overall system.

### 3.1 Preprocessing of noisy text

Preprocessing of noisy texts is done to eliminate the following errors

1. Irrational use of only lower or upper case letters – Figure 1 shows sample text from a blog which uses all lower case letters. Since proper use of lower and upper case letters is necessary for the parser to work correctly, we have employed a regular expression based approach to identify sentence boundaries in noisy texts. Punctuation, capitalization, end-of-line and space characters are used to identify sentence splitting position. To merge, character case and space characters are considered. This module is tuned to work on different data like email, general text and blogs differently, because the pattern can vary across the form of text data. Figure 2 shows the cleaned sentences as extracted from the text shown in figure 1. Noisy data correction for email data is shown in figure 3 and figure 4.

**Figure 1:** Car-blog text, whole text in lower case

```
i have purchased aveo1.4 in july 2007.so far i have driven
over 6000kms.in city it is giving a low mileage of 9to10 km
but on highway it is giving 14+ with a/c on.it is smooth
car.fine riding comfort.gears r smooth.a very silent engine
even with ac.interiors r very attractive.
```

**Figure 2:** Figure 1 text corrected

```
I have purchased aveo1.4 in july 2007.

So far I have driven over 6000kms.

In city it is giving a low mileage of 9to10 km but on
highway it is giving 14  with a/c on.

It is smooth car.

Fine riding comfort.

Gears r smooth.

A very silent engine even with ac.

Interiors r very attractive.
```

**Figure 3:** Noisy email text from enron dataset

```
You're sure to find the perfect gifts that =
will land you on everyone's 'Nice' list. Click here to
redeem your Reward P=
oints now! =09
=09
-------------------------------------------------=
-------------------------------------------------=
------------------------- =09
=09
Forgot your member name??? It is: Allen Forgot your iWon
password? Click here=
:   You received this email because when you registered at
iWon you agreed =
to receive email from us.
```

**Figure 4:** Cleaned and sentence identified from Figure 3 text

```
You're sure to find the perfect gifts that will land you on
everyone's 'Nice' list.
Click here to redeem your Reward Points now!
Forgot your member name?
It is: Allen Forgot your iWon password?
Click  here:  You  received  this  email  because  when  you
registered at iWon you agreed to receive email from us.
```

2. Irrational use of punctuation marks and other symbols – online text is often smeared with symbols and characters which are inserted without much rationale. Our preprocessing module uses some of these symbols to identify segments. Some commonly used expressions in this category are "----, …., ?????" etc.
3. Handling erroneous spelling – By far this is the most important step in the task of opinion mining. Since opinion mining depends on spotting and interpretation of opinion words and opinion features, identifying these in spite of spelling mistakes holds an important key to the accuracy of the system. The challenge here lies in differentiating between two kinds of spelling mistakes:
    (a) The mistake results in a non-dictionary word – this is still easier to tackle by replacing it with the most probable correct word from the dictionary
    (b) The mistake results in another dictionary word with wrong usage. For example, in the following sentence extracted from a car blog, "*the car is really god*", "*god*" is a dictionary word, but in the current context, it is actually "*good*" that is mis-spelt as "*god*". Our aim is to identify and correct this error.

We have employed a statistical approach to correct erroneous spellings, based on suggestions provided by *Suggester (http://www.softcorporation.com/products/suggester)* - a freely available java spell checker. *Suggester* suggests a number of possible variants of a word with their ranks. If the word is a correct dictionary word then it gives that word on the top of the suggestion list with rank 0. For mis-spelt words, however the top-most word may not always be the best choice as shown with the example above. In our system, we have employed a *weighted function based on domain-frequency* of a word to suggest the correct spelling of a possibly mis-spelt word. The other features used for identifying the most probable word for any given word are its left and right neighbors. For a given word *w*, the best possible correct domain word is identified as follows.

For each *w* given by suggester, weight of *w* is calculated by adding four factors. *Rank(w)* is the rank of suggestion word *w* given by *Suggester*. *Freq(w)* is the frequency of *w* in domain text data collection. ***t*** is an integer threshold value specified by us. This is generally close to the maximum *freq(w)* for all non-stopwords *w*. *left(w)* and *right(w)* is left and right word respectively. For example in sentence, *". . . the care is good . . ."*, if spelling of *"care"* is being checked then value of *left(word)* and *right(word)* will be *"the"* and *"good"* respectively. *freq(w1,w2)* is the frequency of word pair <w1,w2> as consecutive word sequence in domain data collection. Based on these values, we define four different weights – *IR(w) - inverse rank of w, DF(w) domain frequency of w, LA(w) - left association of w* and *RA(w) - right association of w.*

$$IR(w)=1/(1+rank(w))$$

$$DF(w)=\{\, freq(w)/t,\ \text{when } freq(w)<=t$$

$$1,\ \text{when } freq(w)>t\}$$

$$LA(w,left(w))=(freq(left(w),w))/(freq(w))$$

$$RA(w,right(w))=(freq(w,right(w)))/freq(w)$$

In above equations *w* is the current suggestion word. Final weight of a suggestion word *w* is calculated as follows. The linear combination of following parameters has shown a good result so we worked with it. Also we can give some weight to individual component to increase or decrease effect of the same.

$$final\_weight(w)=IR(w)+DF(w)+LA(w,left(w))+RA(w,right(w))$$

Suppose a *suggestion vector S={w0, w1, . . ., wn}* and corresponding *rank vector R={r0, r1, . . ., rn}* given by *suggester* for a word *w*. Then *w* will be replaced by $w_i$ *when* $final\_weight(w_i)=MAX_{(j=(0,n))}(final\_weight(w_j))$

This approach works well. For example *"the cae is god"* has been corrected as *"the car is good"*, and when the text is like *"oh god"* then *god* is not changed as *good.*

Figure 5 shows a piece of blog text with spelling mistakes and figure 6 shows how they are corrected using our approach.

**Figure 5:** Car-review text – with spelling mistakes (*depserately, headl ights, dashborad)*

```
The engine still falls short of expectations.
It needs another 1000 CC capacity to decently pull
the 1.7 ton monster.
Still it is an improvement over the earlier lumbering
thing.
But what it depserately needs is a better A/C, better
headl ights and horn plus better spare wheel cover.
The dashborad still feels tacky.
```

**Figure 6:** Car-review text – after spell check

```
The engine still falls short of expectations.
It needs another 1000 cc capacity to decently pull
the 1.7 ton monster.
Still it is an improvement over the earlier lumbering
thing.
But what it desperately needs is a better A/C, better
head lights and horn plus better spare wheel cover.
The dashboard still feels tacky.
```

## 3.2 Generating Opinion Words

The system is initially trained using a semi-supervised approach to generate a larger set of opinion words. It is bootstrapped with a small seed set of basic opinion words selected from the dictionary. This set includes words from both positive and negative categories and are selected from among the most frequently occurring adjectives in a representative domain repository. We choose a mix of pre-processed noisy text and clean text for this purpose. We have tested the algorithm to generate opinion words for various domains by selecting both domain specific and domain-agnostic key words. A judicious mix of the two however is found to give best results. The set can be further expanded at run-time through the module “iterativelyExpand”, which allows the system to learn more application-specific words, by tuning the parameters automatically.

The root words are expanded to identify a larger set of opinion words, using Princeton Wordnet 2.0. A new algorithm has been developed for this purpose. This algorithm considers each synonym of an opinion word, and computes its probability of belonging to both positive and negative categories. The probability of a word belonging to a particular category is dependent on the number of senses in which it overlaps with the total set of synonyms found in each category. An intelligent implementation has been done to extract a large set of opinion words efficiently, with both time and memory requirements optimized. For example by taking *good* as a seed word with weight 1 we are getting words like *solid, worthy, fine* etc. with weights 0.3, 0.8, 0.5 respectively. At later stage words like *advantageous* is added dynamically with weight 1. We present here the results for identifying product related opinion words, irrespective of the product. Starting with 19 positive words, and 10 negative words, this method identifies 679 positive words and 515 negative words with high level of accuracy. The trade-off between accuracy and number of words obtained after expanding can be further tuned by changing some parameters, like the number of levels to consider while expanding WordNet, or the weight to be assigned to level etc.

**Word expansion algorithm:**

```
Expand seed words using Wordnet

Method: expandWord

Input: Initial Word-Set for each category

Output: Extended word-set and weightage for each
words for each category

Steps:
1. call expandWordCat() for each category cat
2. output of expandWordCat() is written into file

-------------------------------------------------
Method:  expandWordCat

Inputs:
cat - category
wordList - basic word list for category-cat
probList - weight of words for category-cat
depth - number of level to look for new words

Outputs:
extWordList - extended word list
extProbList - extended probability list

Steps:
1. include all elements of wordList into
extWordList
2. for i:=0 to i<size(wordList) repeat 2 to 9
  3. w:=wordList(i)
  4. synSet:=synonym(w)
  5. for j:=0 to j<size(synSet) repeat 6 to 7
    6. if synSet(j) has a synonym same as w then
   7. add synSet(j) into extWordList and
probList(i) into extProbList
  8. for d:=0 to d<depth do 8
  9. call iterativelyExpand()
10. call setProbability()

-------------------------------------------------
Method: iterativelyExpand

Inputs:
       extWordList - list of words
       extProbList - list of weights of words
       d - current depth
       t1 - threshold 1
       t2 - threshold 2

Output:
       modified extWordList and extProbList
Steps:
1. oldExtWordList:=extWordList
2. for i:=0 to i<size(oldExtWordList) repeat 3 to
7
   3. word := oldExtWordList(i)
   4. n := number of sense of word from WordNet
   5. m := number of sense of word w is matched
   6. a sense is matched when its t1 fraction of
senseWords are matched
   7. a senseWord is matched when
      (it all ready exist in oldExtWordList) or
      (a synonym of this word exist in
      oldExtWordList with probability > t2)

8. include non existing words of matched sense
into extWordList
9. update probabilities of each word with
prob(word):=m/n
-------------------------------------------------
Method: setProbability

Description: extProbList contains independent
probability of each word
this method will calculate probability of an word
with respect to its parent

Note: parent(w) is the word from which word w is
obtained as a synonym

Inputs:
       extProbList
       extWordList

Output:
       finalProbList

Steps:
1. for each word w in extWordList
2. find the parent(w) such that parent(w) has
maximum probability among all possible parents
3. prob(w):=prob(w)*prob(parent(w))
4. add prob(w) to finalProbList
5. return finalProbList
```

Figure 7 explains the rationale of the algorithm. Let *A* be the word-set we got from last iteration. Now each word of *A* is used to get synonyms using Wordnet. In synonym-set, a word *w* is said to be matched when *w* belongs to *A* or any synonym of *w* belongs to *A* with high probability. A sense is called matched when its *t1* percentage of word is satisfying previously mentioned word matching condition. If a sense *S* is matched the word-set *A* is updates to $A \cup S$. Finally the probability of a word *w* is set by

*Prob(w):=Prob(parent(w))*(no. of matched senses of w / total no. of senses of w)*

It may be noted that that the sense of a word is used to determine its probable orientation, and no sense extraction is done while processing opinions from text.

**Figure 7:** Word matching condition

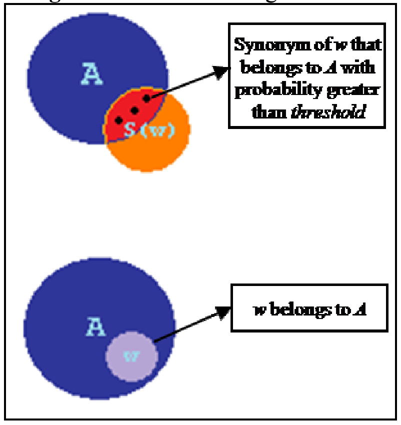
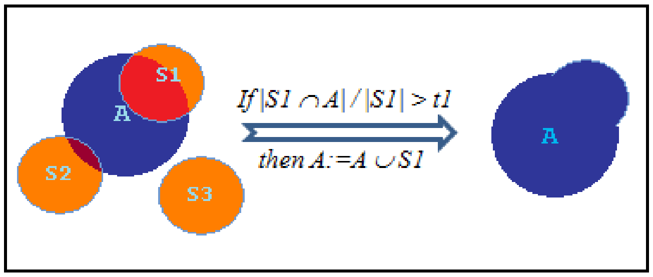

A
S(w)
Synonym of w that belongs to A with probability greater than threshold
A
w
w belongs to A

**Figure 8:** New words inclusion



## 3.3 Identifying opinion expressions and opinionated features from pre-processed noisy text

As already stated, opinion mining consists of three tasks (a) extracting opinion expressions from text (b) analyzing them to extract opinion words and opinionated features (c) classify opinion according to orientation. Opinion expressions are identified and analyzed using Natural Language Processing Tools like Taggers and Parsers. The steps are as follows:

(i) Opinion expressions in a sentence are identified as those which are either adjective phrases, or fragments of a sentence that contain an adjective. Opinion Feature algorithm described below analyzes opinion expressions to identify opinionated words and opinion features. It uses linguistic attributes of the text from sentence fragments. A knowledge-based approach is used to analyze the phrase structure tree, and dependencies among words as extracted by Stanford Parser. Knowledge about auxiliary verbs and prepositions is used to construct rules to identify the opinionated expressions. The rule set presently exploits adjectives as the key to identify opinion expressions. It is being extended to use verbs and noun.

(ii) Opinion word orientations are used to determine orientation of opinion about opinion features. This also exploits a knowledge based approach.

**Opinion feature extraction algorithm**

**Identifies opinion words and opinion features within opinion expressions**

```
Method: extractFeature

Input:
        a sentence from pre-processed text

Output:
        adjectives - all adjectives
        leftFeatures - opinion features before adjectives
        rightFeatures - opinion features after adjectives
        complement - prepositional complements of adjectives
        complement_body - complement text corresponding to complement

Steps:
1. parse sentence to get phraseTree and depTree
2. iterate through phraseTree
3. if ADJP encountered then
   3.1. adjPhrase:=current adjective phrase
   3.2. adjectives:=extract adjective from adjPhrase  /* These are candidate opinion words*/
   3.3. adjpLeftSib:=left sibling of adjPhrase in phraseTree
   3.4. if adjpLeftSib is NP then
        3.3.1. leftFeature(adjPhrase):=adjpLeftSib
   3.5. else
        3.5.1. vpParent:=VP parent of ADJP
        3.5.2. leftFeature:=left NP sibling of vpParent
        3.5.3. if vpParent is not an auxiliary verbs then
               3.5.3.1. add vpParent to leftFeature List
   3.6. rightFeature:=right noun phrase or noun sibling of adjPhrase
```

```
    3.7. from prepositional dependency get
prepositional complements
    3.8. for each complement, complement's children
in p hrase tree is complement body
4. if JJ.* encountered and this is not direct
child of ADJP then
    4.1. adjectives:=word for JJ.* tag /* These are
candidate opinion words */
    4.2. next continuous adjectives and nouns are
concatenated to form right feature
```

Opinion feature extraction algorithm extracts opinion features and stores them as either left-feature or right-feature with respect to the candidate opinion word. The adverbial modifiers and prepositional complements are identified. These are used to decide the orientation of the opinions as described in the next section. For example, from the sentence "*Battery life is very good", "Battery life"* is identified as the opinion feature, the opinion word that is describing *battery life* is *"good"*. The word *"very"* is identified as the adverbial modifier of opinion *"good"*. Similarly in sentence *"the design is sleek and the color screen has good resolution", "design"* is the left feature for *"sleek"* while *"resolution"* is right feature for *"good"*. In sentence *"Good AC, good steering, gearbox and very comfortable to drive",* the opinion "*very comfortable"* is complemented by *"to drive"* .

In the next section we will describe the opinion orientation assignment to the opinions. This will help user to know what the reviewer has said about that feature of a specific product.

## 3.4 Deciding the orientation of the opinion

The candidate opinion words in the opinion features are now to be assigned memberships to positive, negative or neutral orientation classes. The following possibilities are considered:

(i) If a candidate opinion word belongs to the generated set of opinion words (as described in section 3.2), the initial memberships are assigned from that list. The matching takes into account morphological variations of the opinion words.
(ii) If the word does not belong to generated list, its occurrence frequency is observed and if it is above a specified threshold, "iterativelyExpand" is run to compute its orientation values, based on its distance to the words already existing in the expanded set. An example in this category is the word "*plush*", which occurs frequently in car blogs. Though not in the original set of opinion words, this word gets included in the set with a higher positive membership, due to its closeness to words like "rich", "plenty" etc.

Additional linguistic rules are now employed to derive the final orientation of the opinion expressions. Adverbial and negation modifiers of the opinion word are considered to modify the weight and orientation of that word. For example, expressions containing *"good", "very good"* and *"not so good"* will have different scores assigned by this method. Adverbial modifiers can act either as intensifiers, or as decelerators, or as negators. Accordingly they increase, decrease or negate the orientation of membership values to different categories. When negation modifier is associated with the opinion word then its effect is reversed. For example *"picture quality is not good"* will mean that picture quality is bad.

We have considered six categories of adverbial modifiers as depicted in Table 1.

**Table 1:** Adverbial Modifier groups

| | |
|---|---|
| Group 1 – Very high intensifiers | Intensely, extremely, . . . |
| Group 2 - High intensifiers | Very, too, . . . |
| Group 3 – Mild decelerator | Quite, almost, . . . |
| Group 4 – High decelerator | Lightly, mildly, . . . |
| Group 5 – Very high decelerator | Hardly, scarcely, . . . |
| Group 6 - Negator | Not |

## 3.5 Aggregation and assimilation of opinion

The opinion mining system described above produce results in the basic form of opinion words and opinion features. We now present a mechanism to aggregate opinions about features and products at multiple levels of specificity. It may be observed that a feature can be mentioned in different ways within free form text. For example, opinionated features like "*good mileage*, *high fuel efficiency*, *gives good average*" all refer to the basic feature "*mileage*". Our system integrates a domain-specific ontology to aggregate feature level opinions. Table 2 shows different feature expressions that were collected from blogs on cars. The weights indicate normalized collective positive and negative opinion about each feature. Table 3 shows an aggregation of these features with the help of car domain ontology to a core set of three features. The opinion orientations are also suitably summarized.

**Table 2:** Features with opinion before aggregation

| **Feature** | **Positive Opinion Weight** | **Negative Opinion Weight** |
|---|---|---|
| Car | 0.389 | 0.217 |
| Vehicle | 1 | 0.4 |
| Citycar | 1 | 0 |
| Pickup | 0.444 | 0 |
| Acceleration | 1 | 2 |
| Mileage | 0.5 | 0.325 |
| Fuel Efficiency | 1 | 0 |

**Table 3:** Features with opinion after ontology based aggregation

| **Feature** | **Positive Opinion Weight** | **Negative Opinion Weight** |
|---|---|---|
| Car | 0.660555556 | 0.253888889 |
| Pickup | 0.629333333 | 0.666666667 |
| Mileage | 0.611111111 | 0.252777778 |

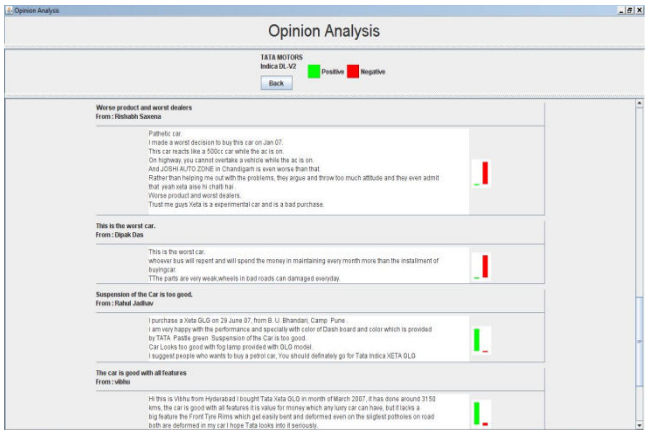
Summarization at the level of a product is similarly obtained by aggregating opinions about all features. Summarization at brand level, repository level, or site levels are also done in a similar way.

## 4. IMPLEMENTATION

The proposed opinion mining system has been developed in JAVA as a web application. The system can be tuned to work on different products with appropriate domain ontology plugged in. The system crawl documents from pre-defined web sites. This can fetch the actual review or feedback text from some widely used websites. Opinionated features and opinions are then extracted at different levels of granularity. Sample features identified from the corpus are [happy owner], [good mileage], [hard gear], [heavy braking], [good pickup], [worse interiors] etc.

Producing results is not enough, unless these can be visualized effectively. Users interact with the system through a Graphical User Interface.

**Figure 9:** Opinion orientation of user feedbacks on a particular brand – each feedback has two scores



**Worse product and worst dealers**
**From : Rishabh Saxena**

Pathetic car.
I made a worst decision to buy this car on Jan 07.
This car reacts like a 500cc car while the ac is on.
On highway, you cannot overtake a vehicle while the ac is on.
And JOSHI AUTO ZONE in Chandigarh is even worse than that.
Rather than helping me out with the problems, they argue and throw too much attitude and they even admit that yeah xeta aise hi chalti hai .
Worse product and worst dealers.
Trust me guys Xeta is a experimental car and is a bad purchase.

**This is the worst car.**
**From : Dipak Das**

This is the worst car.
whoever bus will repent and will spend the money in maintaining every month more than the installment of buyingcar.
TThe parts are very weak,wheels in bad roads can damaged everyday.

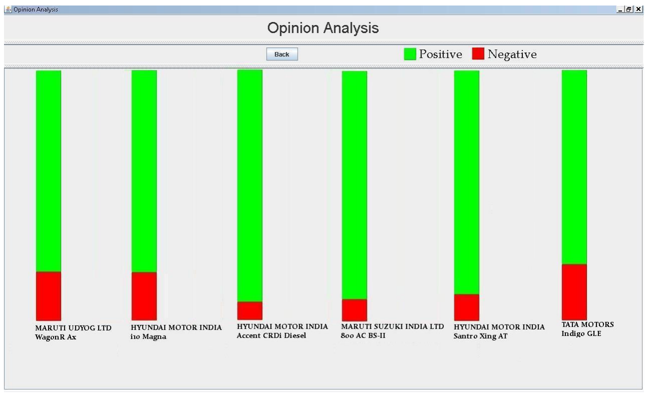
**Suspension of the Car is too good.**
**From : Rahul Jadhav**

I purchase a Xeta GLG on 29 June 07, from B. U. Bhandari, Camp Pune .
I am very happy with the performance and specially with color of Dash board and color which is provided by TATA Pastle green Suspension of the Car is too good.
Car Looks too good with fog lamp provided with GLG model.
I suggest people who wants to buy a petrol car, You should definately go for Tata Indica XETA GLG

**The car is good with all features**
**From : vibhu**

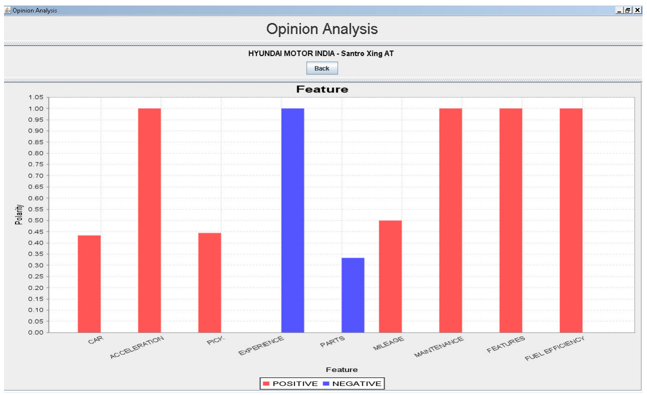
Hi this is Vibhu from Hyderabad I bought Tata Xeta GLG in month of March 2007, it has done around 3150 kms, the car is good with all features it is value for money which any luxry car can have, but it lacks a big feature the Front Tyre Rims which get easily bent and deformed even on the sligtest potholes on road both are deformed in my car I hope Tata looks into it seriously.

**Figure 10:** Comparing different brands based on collective feedback of customers for each brand



Opinion Analysis
Opinion Analysis
Back
🟩 Positive 🟥 Negative
MARUTI UDYOG LTD WagonR Ax
HYUNDAI MOTOR INDIA i10 Magna
HYUNDAI MOTOR INDIA Accent CRDi Diesel
MARUTI SUZUKI INDIA LTD 800 AC BS-II
HYUNDAI MOTOR INDIA Santro Xing AT
TATA MOTORS Indigo GLE

**Figure 11:** Collective opinion orientation of individual features of a brand are presented as positive or negative (Detail Level)



We have presented some screen-shots for analyzing feedbacks provided by users at *indiacar.com*. Opinions are presented to the user brand-wise, model-wise, or feature-wise. It is also possible to view overall positivity or negativity of each user. Figures 9, 10 and 11 show how results are displayed at different levels of granularity.

## 5. EVALUATION

We have evaluated the system in terms of the following capabilities: (1) identifying opinion orientation of words, (2) identifying the features described by these words, (3) assigning positive and negative scores to sentences and (4) measuring overall positive and negative contents at individual user levels and collective levels considered brand-wise. The performance is measured using standard evaluation measures of precision, recall, and F-score. All scores presented here are based on manual evaluation of the corpus done by a team of researchers not involved with this work. This was done to ensure unbiased evaluation.

The evaluations presented here are based on results extracted from 130 customer feedbacks taken from http://www.indiacar.com for 10 models of car. 810 sentences were manually tagged for opinion orientation, in which 1043 words were identified as possible opinion words by human evaluators. Repetition was there e.g. *good* appears 48 times. Among 130 feedbacks 88 were judged by human evaluators as overall positive, 36 were tagged as overall negative and 6 were labeled as neutral. Total 966 repetitive features were manually identified.

The system was initialized with a seed set of 29 words, which were assigned membership values of 0 or 1 to categories "positive" and "negative". Each seed word belonged to exactly one category with membership 1. The seed set was then expanded using WordNet and a total of 1194 words were identified as opinion words. A single word may appear in both positive and negative category. Opinion mining from customer feedbacks was then initiated with this set. The aim was to evaluate the capability of the system to recognize as many of the 1043 words from the reports correctly as possible. Out of these 1043 words, 813 were identified as exact or stemmed matches with the expanded set. The feedback reports were then subjected to POS analysis. The *iterateexpansion* phase is then initiated to recognize more adjectives as opinion words, with a more relaxed constraint. Table

4 presents precision and recall values for detecting opinion words from the feedbacks, at various levels. High precision value in Table 4 shows that the proposed algorithm to find opinion words through WordNet works well. The Recall value can be improved further by including more domain-dependent words in the root set. Table 5 presents evaluation of the system in identifying opinionated features and their opinion orientations correctly. The recall and precision values shown in this table reflect the accuracy of the linguistic rules implemented for opinion feature identification. High scores in this table imply that the linguistic rules work fairly well in spite of noisy data, since our opinion expression algorithm mainly uses localized dependencies.

Table 6 presents results on system performance in categorizing opinions both at sentence level and report level. The report level summarization averages and normalizes the sentence level values.

This is due to the fact that only adjectives are being considered presently. This is likely to go up when verbs are also included in the system. Table 6 also shows the performance of the system at individual feedback level. This is the collective score assigned for all sentences. The precision rises from 0.88 to 0.94. Recall value also improves from 0.48 to 0.75. User level feedbacks are aggregated brand-wise to produce collective feedback.

**Table 4:** Accuracy of identifying orientation of opinion words

| Recall | Precision | F-Score |
|---|---|---|
| 0.78 | 0.94 | 0.85 |

**Table 5:** Accuracy of feature extraction from opinion expressions

| Recall | Precision | F-Score |
|---|---|---|
| 0.95 | 0.91 | 0.93 |

**Table 6:** Accuracy of assigning opinion scores

| | Recall | Precision | F-Score |
|---|---|---|---|
| At Sentence levels | 0.48 | 0.87 | 0.62 |
| User level (all sentences) | 0.75 | 0.94 | 0.83 |

## 6. Conclusion

In this paper we have presented an opinion mining system that can extract opinions about products and features from noisy texts like blogs, emails, on-line customer feedback etc. The system uses a plugged in domain ontology to extract opinions from pre-defined web sites. Opinions can be viewed at multiple levels of granularity based on user requirement.

The key challenge to mine opinion from such texts is posed by the fact that the text is noisy. We have proposed a text pre-processing mechanism which exploits domain knowledge to clean the text. The cleaned text is then processed by Natural Language Processing tools. We have also proposed an iterative approach to integrate new knowledge into the system. The system needs minimum user intervention and can be tuned for different domains. User involvement is limited to choosing an appropriate domain ontology and selection of configuration parameters to bootstrap the system.

We have presented novel algorithms to determine orientation of opinion words using *Wordnet.* We are currently working towards increasing the scope of opinion expression identification. We are also enhancing our system to learn context dependent opinion orientation from a training set. For example, a word like "*comfortable*" can always be judged as having positive orientation independent of domain or context. However words like "*big, small, etc.* ", cannot be classified as "positive" or "negative" in a context independent way. In a sentence like "*the screen is small for the camera*", *small* should be interpreted to have a negative connotation. However, when it is mentioned "*the camera is small enough to fit into the pocket*", it should be interpreted to have a positive connotation. This can be done provided the system learns that "*big*" is desirable for "*screen*" but "*small*" is desirable for "*size*", when it comes to the domain of camera. We need to have appropriate annotated training sets however to do this.

## 7. REFERENCES

[1] Pavel Smrž, "Using WordNet for Opinion Mining", GWC 2006 Proceedings, pp. 333–335, 2006

[2] Tetsuya Nasukawa, Diwakar Punjani, Shourya Roy, L. Venkata Subramaniam, Hironori Takeuchi, "Adding Sentence Boundaries to Conversational Speech Transcriptions using Noisily Labelled Examples", Proc. IJCAI-2007 Workshop on Analytics for Noisy Unstructured Text, pp. 71-78, India, 2007

[3] Wilson Wong, Wei Liu and Mohammed Bennamoun, "Integrated Scoring for Spelling Error Correction, Abbreviation Expansion and Case Restoration in Dirty Text", Proc. AusDM2006 Fifth Australasian Data Mining Conference, CRPIT Volume 61, pp. 83-89, Sydney, 2006

[4] Wilson Wong,Wei Liu and Mohammed Bennamoun, " Enhanced Integrated Scoring for Cleaning Dirty Texts", Proc. IJCAI-2007 Workshop on Analytics for Noisy Unstructured Text, pp. 55-62, India, 2007

[5] Minqing Hu and Bing Liu, "Mining and Summarizing Customer Reviews", KDD'04, Washington, 2004

[6] Minqing Hu and Bing Liu, "Mining Opinion Features in Customer Reviews", American Association for Artificial Intelligence, 2004

[7] Andrea Esuli and Fabrizio Sebastiani, "SENTIWORDNET: A Publicly Available Lexical Resource for Opinion Mining", LREC2006 Conference on Language Resources and Evaluation, Genova, 2006

[8] A-M. Popescu and O. Etzioni, "Extracting Product Features and Opinions from Reviews", EMNLP-05, Canada, 2005