Opinion-Based Imprecise Query Answering

Muhammad Abulaish[1,*], Tanvir Ahmad[2], Jahiruddin[1], and Mohammad Najmud Doja[2]

[1] Department of Computer Science, Jamia Millia Islamia, New Delhi, India
[2] Department of Computer Engineering, Jamia Millia Islamia, New Delhi, India
{abulaish.cs,tanvir.ce}@jmi.ac.in,
jahir.jmi@gmail.com, ndoja@yahoo.com

Abstract. There is an exponential growth in user-generated contents in the form of customer reviews on the Web. But, most of the contents are stored in either unstructured or semi-structured format due to which distillation of knowledge from this huge repository is a challenging task. In addition, on analysis we found that most of the users use fuzzy terms instead of crisp terms to express opinions on product features. Considering these facts, in this paper, we present an opinion-based query answering framework which mines product features and opinionated words to handle user queries over review documents. The proposed framework uses BK-FIRM (Bandler-Kohout Fuzzy Information Retrieval Model) that facilitates the formulation of imprecise queries using linguistic qualifiers, retrieves relevant opinion documents, and presents them in the order of their degree of relevance. The efficacy of the system is established through experiments over customer reviews on different models of digital camera, and mp3 player.

Keywords: Opinion Mining, Sentiment Analysis, Opinion-Based Query Answering, Imprecise Query Processing, Natural Language Processing.

1 Introduction

Due to easy accessibility of Web, numerous forums, discussion groups, and blogs exist and individual users are participating more actively and are generating vast amount of new data – termed as *user-generated contents*. These new web contents include customer reviews and blogs that express opinions on products and services – which are collectively referred to as customer feedback data on the Web. As customer feedback on the Web influences other customer's decisions, these feedbacks have become an important source of information for businesses to take into account when developing marketing and product development plans. Now much of the information is publicly available on the Web. As a result, the number of reviews that a product receives grows rapidly. Some popular products can get hundreds of reviews or more at some large merchant sites. Many reviews are also long, which makes it hard for potential customers to read them to make an informed decision on whether to purchase the product. A large number of reviews for a single product may also make

[*] Corresponding author.

it harder for individuals to evaluate the true underlying quality of a product. In these cases, customers may naturally gravitate to reading a *few reviews* in order to form a decision regarding the product and he/she only gets a biased view of the product. Similarly, manufacturers want to read the reviews to identify what elements of a product affect sales most. And, the large number of reviews makes it hard for product manufacturers or business to keep track of customer's opinions and sentiments on their products and services. Recent work has shown that the distribution of an overwhelming majority of reviews posted in online markets is bimodal [7]. Reviews are either allotted an extremely high rating or an extremely low rating. In such situations, the average numerical star rating assigned to a product may not convey a lot of information to a prospective buyer. Instead, the reader has to read the actual reviews to examine which of the positive and which of the negative aspect of the product are of interest. Several sentiment analysis approaches have proposed to tackle this challenge up to some extent. However, most of the classical sentiment analysis mapping the customer reviews into binary classes – *positive or negative*, fails to identify the product features liked or disliked by the customers.

In this paper, we present an opinion-based query answering framework that mines product features and opinionated words from opinion texts. The proposed framework uses BK-FIRM (Bandler-Kohout Fuzzy Information Retrieval Model) to handle opinion-oriented imprecise user queries over review documents. Linguistic and semantic analyses are applied to identify key information components that are centered on product features. Since, on analysis we found that most of the users use fuzzy terms instead of crisp terms to express opinions on product features, an information component is defined as a triplet $<\mathcal{F}, \mathcal{M}, \mathcal{O}>$ where, $\mathcal{F}$ represents a product feature, $\mathcal{O}$ represents opinion words associated with $\mathcal{F}$ and $\mathcal{M}$ is an optional component representing adverbs that act as modifier and used to intensify the opinion $\mathcal{O}$. $\mathcal{M}$ is also used to capture the negative opinions explicitly expressed in the review. The novelty of the system lies in mining associated modifiers with opinions. For example, consider following snippets of opinion sentences: *(i) the picture quality is very good; (ii) the picture quality is almost good.* In both of the sentences the opinion word is *good* but the associated modifiers are different to express different levels of customer satisfaction on picture quality. For each extracted feature, the list of opinions and associated modifiers are compiled and stored in a structured repository to answer user query over it.

The remaining paper is structured as follows: Section 2 presents a brief review on opinion mining. It also presents the overview of the BK-FIRM model and its working principles. In section 3, we present the opinion-based query answering framework. The experimental setup and evaluation results are presented in section 4. Finally, section 5 concludes the paper with possible enhancements to the proposed system.

2 Related Work

In this section, we present a summarized view of the existing works on opinion mining and sentiment analysis which is followed by a brief introduction of the BK-FIRM model and its working principles.

2.1 Opinion Mining and Sentiment Analysis

Research on opinion mining started with identifying opinion bearing words, e.g., *great*, *amazing*, *wonderful*, *bad*, *poor* etc. In literature, a reasonable number of attempts have been made to mine such words and identifying their semantic orientations [3,5]. The history of the phrase *"sentiment analysis"* parallels that of *opinion mining* in certain respects. A sizeable number of papers mentioning sentiment analysis focus on the specific application of classifying customer reviews as to their polarity – positive or negative [10,11]. Although, classical sentiment classification attempts to assign the review documents either *positive* or *negative* class, it fails to find what the reviewer or opinion holder likes or dislikes. A positive document on an object does not mean that the opinion holder has positive opinions on all aspects or features of the object. Likewise, a negative document does not mean that the opinion holder dislikes everything about the object. In an evaluative document (e.g., *a product review*), the opinion holder typically writes both positive and negative aspects of the object, although the general sentiment on the object may be positive or negative. To obtain detailed aspects, feature-based opinion mining is proposed in literature [3,4,6]. In [4], a supervised pattern mining method is proposed. In [3, 6], an unsupervised method is used. A lexicon-based approach has been shown to perform quite well in [2, 3]. The lexicon-based approach basically uses opinion words and phrases in a sentence to determine the orientation of an opinion on a feature.

Although, some opinion mining methods extract features and opinions from document corpora, most of them do not explicitly exploit the semantic relationships between them. The proposed method differs from all these approaches predominantly in its use of pure linguistic techniques to identify only those features for which customers have commented using opinionated words. Extraction of associated modifiers used in review documents to represent the degree of expressiveness of opinions is unique in our work. Moreover, to the best of our knowledge, none of the above-cited works attempted to use the mined features and opinions for query answering.

2.2 BK-FIRM

Different from traditional information retrieval theories, BK-FIRM uses the concept of fuzzy relation to retrieve documents based on semantics and it has basic functions such as automated building of a thesaurus and ranking the retrieved documents. BK-FIRM has two operations, (i) R-request operation which expands semantics of a term, and (ii) FS-request operation which analyzes user query and retrieves documents relevant to the given query [1]. The procedure of BK-FIRM is as follow. Assume that there are a document set $D=\{d_1, d_2, \ldots, d_k\}$, a term set $T=\{t_1, t_2, \ldots, t_n\}$ and a fuzzy relation $\mathfrak{R}_F$ (equation 1) between the document set and the term set. When query Q as FS-request is given from a user, the fuzzy relation $\mathfrak{R}_F$ applied to the query Q gets a fuzzy set D_F (equation 2), which means the suitability of the query Q to document set. Then, an α-cut is applied to the fuzzy set D_F to get a resultant document set D_R (equation 3).

$$\Re_F = D \times T = \begin{array}{cccc|c} \begin{bmatrix} v_{11} & v_{12} & \cdots & v_{1n} \\ v_{21} & v_{22} & \cdots & v_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ v_{k1} & v_{k2} & \cdots & v_{kn} \end{bmatrix} \begin{matrix} d_1 \\ d_2 \\ \vdots \\ d_k \end{matrix} \\ \begin{matrix} \mathrm{t}_1 & t_2 & \cdots & t_n \end{matrix} \end{array} \tag{1}$$

$$D_F = Q(\Re_F) = Q(\begin{array}{c} \begin{bmatrix} v_{11} & v_{12} & \cdots & v_{1n} \\ v_{21} & v_{22} & \cdots & v_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ v_{k1} & v_{k2} & \cdots & v_{kn} \end{bmatrix} \begin{matrix} d_1 \\ d_2 \\ \vdots \\ d_k \end{matrix} \\ \begin{matrix} \mathrm{t}_1 & t_2 & \cdots & t_n \end{matrix} \end{array}) = \{d_1/v_1, d_2/v_2, \cdots, d_k/v_k\} \tag{2}$$

$$D_R = \{d_1', d_2', \cdots, d_k'\} \tag{3}$$

3 Proposed Framework

Fig. 1 presents the architectural details of the proposed opinion-based query answering framework which consists of two major modules – *Feature and Opinion Learner, and Opinion-Based Query Processor.* The working principles of these components are explained in the following sub-sections.

3.1 Feature and Opinion Learner

In this section, we present the working details of the feature and opinion learner module which completes its task in the following three steps (i) Document processing and subjectivity analysis, (ii) Document parsing, and (iii) Feature and opinion extraction.

3.1.1 Document Processing and Subjectivity Analysis

We employ *document processing* to divide an unstructured web document into individual record-size chunks, to clean them by removing ML tags, and to present them as individual unstructured record documents for further processing. The cleaned documents are converted into numeric-vectors using unigram model for the purpose of subjectivity analysis. In document vectors a value represents the likelihood of each word being in a subjective or objective sentence.

According to Pang and Lee [9] subjective sentences are expressive of the reviewer's sentiment about the product, and objective sentences do not have any direct or obvious bearing on or support of that sentiment. Therefore, the idea of subjectivity analysis is used to retain segments (sentences) of a review that are more subjective in nature and filter out those that are more objective. This increases the system performance both in terms of *efficiency* and *accuracy*. The idea proposed by Yeh [8]

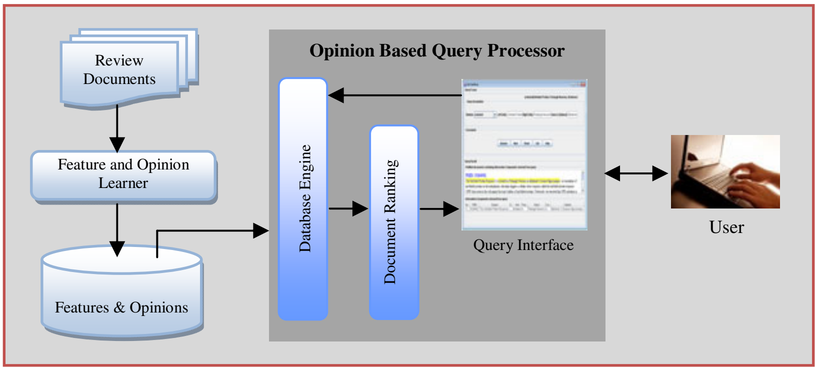


Fig. 1. Proposed opinion-based query answering framework

is used to divide the reviews into subjective parts and objective parts. In [8], the idea of cohesiveness is used to indicate segments of a review that are more subjective in nature versus those that are more objective. We have used a corpus of subjective and objective sentences used in [9] for training purpose. The training set is used to get the probability for each word to be subjective or objective, and the probability of a sentence to be subjective or objective is calculated using the unigram model. The Decision Tree classifier of Weka[1] is trained to classify the unseen review sentences into subjective and objective classes.

3.1.2 Document Parsing

Since our aim is to extract product features and the opinions from text documents, all subjective sentences are parsed using Stanford Parser[2] which assigns Parts-Of-Speech (POS) tags to English words based on the context in which they appear. The POS information is used to locate different types of information of interest inside the text documents. For example, generally noun phrases correspond to product features, adjectives represent opinions, and adverbs are used as modifiers to represent the degree of expressiveness of opinions. Since, it is observed that opinion words and product features are not independent of each other rather, directly or indirectly inter-related through some semantic relations, each sentence is converted into dependency tree using Stanford Parser. The dependency tree, also known as *word-word relationship*, encodes the grammatical relations between every pair of words. A sample POS tagged sentence and the corresponding dependency tree generated using Stanford Parser is shown in Fig. 2(a) and 2(b) respectively.

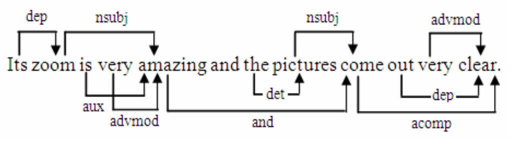
3.1.3 Feature and Opinion Extraction

This process takes the *dependency tree* generated by document parser as input and output feasible information components after analyzing noun phrases and the associated adjectives possibly preceded with adverbs. On observation, we found that

[1] `http://www.cs.waikato.ac.nz/~ml/weka/`
[2] `http://nlp.stanford.edu/software/lex-parser.shtml`

Its/PRP$ zoom/NN is/VBZ very/RB amazing/JJ and/CC the/DT pictures/NNS come/VBP out/IN very/RB clear/JJ ./.

(a) A POS-tagged sentence


dep
nsubj
nsubj
advmod
Its zoom is very amazing and the pictures come out very clear.
aux
advmod
det
and
dep
acomp

(b) Dependency tree

<zoom, very, amazing> // Extracted information component through Rule-1

<pictures, very, clear> // Extracted information component through Rule-2

(c) Extracted Information Components

Fig. 2. (a) A POS-tagged sentence, (b) the corresponding dependency tree generated by Stanford Parser, and (c) extracted information components

product features are generally noun phrases and opinions are either only adjectives or adjectives preceded by adverbs. For example, consider the following opinion sentence:

```
(ROOT(S(NP(NP (DT The) (NN battery) (NN life))(PP (IN of)
(NP (NNP Nokia) (NNP N95))))(VP (VBZ is)(ADJP (RB very)
(JJ good)))(. .)))
```

In the above sentence, “battery life” is a noun phrase and appears as one of the features of Nokia N95 whereas, the adjective word “good” along with the adverb “very” is an opinion to express the concern of reviewer. Therefore, we have defined the information component as a triplet $<\mathcal{F}, \mathcal{M}, \mathcal{O}>$ where, $\mathcal{F}$ is a noun phrase and $\mathcal{O}$ is adjective word possibly representing product feature. $\mathcal{M}$ represents adverb that acts as modifier to represent the degree of expressiveness of $\mathcal{O}$. $\mathcal{M}$ is also used to capture negative opinions explicitly expressed in reviews. The information component extraction mechanism is implemented as a rule-based system which analyzes dependency tree to extract information components. Some sample rules are presented below to highlight the function of the system.

Rule 1: In a dependency tree $\mathcal{T}$, if there exists a $subj(w_i, w_j)$ relation such that POS(w_i) = JJ*, POS(w_j) = NN*, w_i and w_j are not stop-words[3] then w_j is assumed to be a *feature* and w_i as an *opinion*. Thereafter, the relation $advmod(w_i, w_k)$ relating w_i with some adverbial words w_k is searched. In case of presence of *advmod* relation, the information component is identified as $<w_j, w_k, w_i>$ otherwise $<w_j, -, w_i>$.

Rule 2: In a dependency tree $\mathcal{T}$, if there exists a $subj(w_i, w_j)$ relation such that POS(w_i) = VB*, POS(w_j) = NN*, and w_j is not a stop-word then we search for $acomp(w_i, w_m)$ relation. If *acomp* relation exists such that POS(w_m) = JJ* and w_m is not a stop-word then w_j is assumed to be a *feature* and w_m as an *opinion*. Thereafter, the modifier is searched and information component is generated in the same way as in rule 1.

[3] A list of 571 stop-words available at `http://www.aifb.uni-karlsruhe.de/WBS/aho/clustering`

Fig. 2(c) presents two sample information components extracted by applying these rules on the dependency tree shown in figure 2(b). Though a large number of commonly occurring noun and adjective phrases are eliminated due to the design of the information component itself, it is found that further processing is necessary to consolidate the final list of information components and thereby the product features and opinions. During the consolidation process, we take care of two things. In the first stage, since product features are the key noun phrases on which opinions are applied, so a feasible collection of product features is identified using mutual information [12] value calculated using equation (4). In the second stage of analysis, however, for each product feature the list of all opinions and modifiers is compiled that are used later for indexing and query answering purpose.

The mutual information measure, $I(x, y)$, is used to compare the probability of observing x and y *together* with the probabilities of observing x and y *independently*. If there is a genuine association between x and y, then the joint probability $P(x, y)$ will be much larger than $P(x).P(y)$, and consequently $I(x, y) >> 0$. If there is no interesting relationship between x and y, then $P(x, y) \approx P(x).P(y)$, and thus, $I(x, y) \approx 0$. If x and y are in complementary distribution, then $P(x, y)$ will be much less than $P(x).P(y)$, forcing $I(x, y) << 0$. The probabilities $P(x)$ and $P(y)$ are estimated by counting the number of observations of x and y in a corpus, $f(x)$ and $f(y)$, and normalizing by N, the size of the corpus. The joint probabilities, $P(x, y)$, are estimated by counting the number of times that x is followed by y or y is followed by x in a window of 5 words to consider structural relationship, and normalizing by N. In our application, a list of seed opinion words (positives and negatives) is compiled and mutual information value for a feature word with each of them is calculated. If the cumulated sum value for a feature is zero (i.e., the feature is not associated with any seed opinion word) the feature and the corresponding information component is filtered out, otherwise retained. Thereafter, for each retained feature, the list of opinion words and modifiers are compiled from information components and are stored in a structured form. A partial list of product features, opinions, and modifiers extracted from a corpus of 86 customer reviews on *digital camera* (obtained from www.ebay.com) and from 95 review on mp3 player (used in [3]) is shown in table 1.

$$I(x, y) = \log_2 \frac{P(x, y)}{P(x)P(y)} \tag{4}$$

Table 1. A partial list of extracted features, opinions and modifiers for digital camera

Product	Feature	Modifier	Opinion
Digital Camera	picture	not, really, very	beautiful, clear, fantastic, good, great, professional, sharp
	battery	Very	decent, excellent, rechargeable, short, long
	price	---	cheap, excellent, good, great, high
mp3 Player	player	Very	awesome, delicate, perfect, fast, good, great, terrific, large, excellent
	Sound	pretty, very, indeed	excellent, good, wonderful, excellent, great, awesome
	Software	very, somewhat, enough	great, easy, nice, awful, good, smooth, quick, decent, inferior, awesome, installed, bad

3.2 Opinion-Based Query Processor

In this section, we present the query processing mechanism using BK-FIRM model over structured repository of features and opinions extracted by Feature and Opinion Learner module. To apply BK-FIRM in our case, *D* is the set of all review documents under consideration; the set T is generated for each product feature and it contains all opinion words associated with a particular feature. Thus, for each feature, a fuzzy relation matrix $\mathfrak{R}_F$ is generated in which contents are normalized *tf-idf* values. In order to handle queries on multiple features a user can use fuzzy logic connectives such as AND, OR and NOT, and fuzzy quantifiers as defined in equations (5) to (9).

$$a\ OR\ b = \mu(a) \vee \mu(b) = \max(\mu(a), \mu(b)) \tag{5}$$

$$a\ AND\ b = \mu(a) \wedge \mu(b) = \min(\mu(a), \mu(b)) \tag{6}$$

$$NOT\ \ a = \neg\mu(a) = 1 - \mu(a) \tag{7}$$

$$VERY\ \ a = Q_{very}(\mu(a)) = [\mu(a)]^2 \tag{8}$$

$$FAIRLY\ \ a = Q_{FAIRLYy}(\mu(a)) = [\mu(a)]^{1/2} \tag{9}$$

To illustrate this process, a partial view of fuzzy relations $\mathfrak{R}_{picture}$ and $\mathfrak{R}_{price}$ for two camera features *picture* and *price* are shown in equations (10) and (11) respectively. Given a query *Q = VERY(sharp) AND FAIRLY(high),* i.e., *camera with very sharp picture quality and fairly high price*, we get a fuzzy set D_F (equation 12) which represents the suitability between documents and the query. When α-cut = 0.7 is applied, we can get the documents d_1 and d_2 in this order of relevance.

$$\mathfrak{R}_{picture} = \begin{array}{c} \\ d_1 \\ d_2 \\ d_3 \\ d_4 \\ d_5 \end{array} \begin{array}{c} \begin{array}{cccc} beautiful & good & sharp & clear \end{array} \\ \left[\begin{array}{cccc} 0.7 & 0.6 & 0.9 & 1.0 \\ 0.6 & 0.8 & 1.0 & 0.4 \\ 0.5 & 0.7 & 0.8 & 0.7 \\ 0.2 & 0.7 & 0.6 & 0.5 \\ 0.3 & 0.2 & 0.7 & 0.5 \end{array}\right] \end{array} \tag{10}$$

$$\mathfrak{R}_{price} = \begin{array}{c} \\ d_1 \\ d_2 \\ d_3 \\ d_4 \\ d_5 \end{array} \begin{array}{c} \begin{array}{cccc} cheap & good & great & high \end{array} \\ \left[\begin{array}{cccc} 0.4 & 0.3 & 0.6 & 0.8 \\ 0.6 & 0.5 & 0.7 & 0.5 \\ 0.4 & 0.2 & 0.3 & 0.5 \\ 0.3 & 0.8 & 0.7 & 0.6 \\ 0.9 & 0.4 & 0.5 & 0.3 \end{array}\right] \end{array} \tag{11}$$

$$D_F = \{d_1/0.81, d_2/0.71, d_3/0.64, d_4/0.36, d_5/0.49\} \tag{12}$$

4 Experimental Results

In this section, we present the experimental details of the feature and opinion mining process. For subjectivity analysis, we used the subjectivity dataset[4] v1.0 from Cornell for training purpose. The dataset consists of 5000 subjective sentences and 5000 objective sentences. A Java program is written to extract features using unigram model from this dataset and to convert each sentence into equivalent numeric vector where a value represents likelihood of each word being in a subjective or objective sentence. Thereafter, the Decision Tree classifier of Weka is trained to classify the unseen sentences into subjective and objective classes. The accuracy of the classier on 10-fold cross validation is 82%. The data sample used in our work to mine features and opinions for customer reviews summarization consists of 86 review documents on different models of digital camera (Canon: 60, Panasonic: 26) – all obtained from www.ebay.com, and 95 documents on mp3 player used in [3]. The feature and opinion extraction process described in section 3.1.3 was implemented using Java to mine features and opinionated words along with modifiers from the subjective review sentences. Initially, a total of 48 and 227 for *digital camera* and *mp3 player* respectively were extracted out of which only 33 and 151 were retained after feasibility analysis. For each retained feature, the list of both opinions and modifiers were compiled, a partial view of which is shown in table 1, and stored in structured database. Thereafter, queries were processed over this database using BK-FIRM model to extract relevant review documents.

4.1 Evaluation Methods

The performance of the whole system is analyzed by taking into account the performance of the *feature and opinion extraction* process only as it is difficult to provide a performance analysis of the query-processing module, since no benchmark set of queries are available for judging the performance of such a system. Since the information components are finally stored in a database, the system can obviously retrieve all exact matches correctly. When it comes to judging the relevance of answers to fuzzy query, the quality of retrieval is dependent on the similarity computation procedure. For example, it can be seen from the examples cited above that in some cases, the fuzzy Min-Max function seems to be too restrictive, though we have chosen it since this provides a standard way of interpreting AND and OR boolean operators. We refrain from giving any relevance figure for this module, since acceptability of an answer generated is largely dependent on the user's perspective.

We now present a discussion on the performance of the whole system which is analyzed by taking into account the performance of the *feature* and *opinion* extraction process. Since terminology and complex proper names are not found in Dictionaries, an obvious problem of any automatic method for concept extraction is to provide objective performance evaluation. Therefore manual evaluation has been performed to judge the overall performance of the system. For evaluation of the experimental results, we use standard Information Retrieval performance measures. From the extraction results, we calculate the true positive *TP* (number of correct feature-opinion pairs the system identifies as correct), the false positive *FP* (number of incorrect feature-opinion pairs the system falsely identifies as correct), true negative

[4] `http://www.cs.cornell.edu/people/pabo/movie-review-data/`

TN (number of incorrect feature-opinion pairs the system identifies as incorrect), and the false negatives *FN* (number of correct feature-opinion pairs the system fails to identify as correct). By using these values we calculate the following performance measures:

Precision (π): the ratio of true positives among all retrieved instances.

$$\pi = \frac{TP}{TP + FP} \tag{13}$$

Recall (ρ): the ratio of true positives among all positive instances.

$$\rho = \frac{TP}{TP + FN} \tag{14}$$

F1-measure (F_1): the harmonic mean of recall and precision.

$$F_1 = \frac{2\rho\pi}{\rho + \pi} \tag{15}$$

Accuracy (τ): the ratio of sum of TPs and TNs over total positive and negative instances.

$$\tau = \frac{TP + TN}{TP + FP + FN + TN} \tag{16}$$

The values of the above performance measures are calculated for each category of experimental data. In order to present a synthetic measure of performance over all categories, we present the macro-averaged performance which consists in simply averaging the result obtained on each category. Table 2 summarizes the performance measure values for our system in the form of a misclassification matrix. The recall value is lower than precision indicating that certain correct feature-opinion pairs could not be recognized by the system correctly. This is justified since most of the reviewers do not follow grammatical rules strictly while writing reviews due to which the parser fails to assign correct POS tag and thereby correct dependency relations between words. However, almost all identified feature-concept pairs are correct, which leaves scope for enhancing our grammar to accommodate more dependency relations. After analyzing the review documents manually we also found that some review documents contain junk sentences too which opens a new direction of research – *review spam analysis.*

Table 2. Performance evaluation of feature-opinion extraction process

Product Name		**TP**	**FP**	**FN**	**TN**	**Precision (%)**	**Recall (%)**	**F1-measure (%)**	**Accuracy**
Digital Camera	**Canon**	34	03	26	314	91.89	56.67	70.10	92.30
	Panasonic	31	03	17	174	91.18	64.58	75.61	91.11
mp3 player		264	33	149	1287	88.89	63.92	74.37	89.50
					Macro-Average	**90.65**	**61.72**	**73.36**	**90.97**

5 Conclusion and Future Work

In this paper, we have proposed an opinion-based query answering framework which performs linguistic and semantic analysis of text to identify product features and opinions from review documents. We have also proposed a method using BK-FIRM model to handle imprecise user queries, formulated using fuzzy quantifiers, over review documents. Presently, we are refining the rule-set to consider more dependency relations to improve the *precision* and *recall* values of the system. Instead of directly using standard membership functions for fuzzy quantifiers and ignoring the one present in review documents for relevance computation, we are also exploring a fuzzy similarity computation method that would consider both the quantifiers present in user query and in retrieved documents for relevance computation.

References

1. Kohout, L.J., Keravnou, E., Bandler, W.: Automatic Documentary Information Retrieval by Means of Fuzzy Relational Products. In: Gaines, B.R., Zadeh, L.A., Zimmermann, H.J. (eds.) Fuzzy Sets in Decision Analysis, pp. 308–404 (1984)
2. Ding, X., Liu, B., Philip, S.Y.: A Holistic Lexicon-Based Approach to Opinion Mining. In: Proceedings of the first ACM International Conference on Web search and Data Mining (WSDM'08), California, USA, pp. 231–240 (2008)
3. Hu, M., Liu, B.: Mining and Summarizing Customer Reviews. In: Proceedings of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'04), USA, pp. 168–177 (2004)
4. Liu, B., Hu, M., Cheng, J.: Opinion Observer - Analyzing and comparing opinions on the Web. In: Proceedings of the 14th International Conference on World Wide Web (WWW'05), Japan, pp. 342–351 (2005)
5. Kim, S., Hovy, E.: Determining the Sentiment of Opinions. In: Proceedings of the 20th International Conference on Computational Linguistics (COLING'04), Switzerland, pp. 1367–1373 (2004)
6. Popescu, A.M., Etzioni, O.: Extracting Product Features and Opinions from Reviews. In: Proceedings of the 2005 Conference on Empirical Methods in Natural Language Processing (EMNLP'05), Canada, pp. 339–346 (2005)
7. Ghose, A., Ipeirotis, P.G.: Designing Ranking Systems for Consumer Reviews: The Impact of Review Subjectivity on Product Sales and Review Quality. In: Proceedings of the Workshop on Information Technology and Systems (WITS'06), Milwaukee (2006)
8. Yeh, E.: Final Project Picking the Fresh from the Rotten: Quote and Sentiment Extraction from Rotten Tomatoes Movie Reviews, CS224N/Ling237 (2006)
9. Pang, B., Lee, L.: A Sentimental Education: Sentiment Analysis Using Subjectivity Summarization Based on Minimum Cuts. In: Proceedings of ACL'04, pp. 271–278 (2004)
10. Turney, P.: Thumbs Up or Thumbs Down? Semantic Orientation Applied to Unsupervised Classification of Reviews. In: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics (ACL'02), Philadelphia, Pennsylvania, pp. 417–424 (2002)
11. Pang, B., Lee, L., Vaithyanathan, S.: Thumbs up? Sentiment Classification Using Machine Learning Techniques. In: Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP'02), USA, pp. 79–86 (2002)
12. Church, K.W., Hanks, P.: Word Association Norms, Mutual Information, and Lexicography. Computational Linguistics 16(1), 22–29 (1990)