A Frequent Pattern Mining Algorithm for Feature Extraction of Customer Reviews

Seyed Hamid Ghorashi[1], Roliana Ibrahim[2*], Shirin Noekhah[3] and Niloufar Salehi Dastjerdi[4]

**[1] Software Engineering Research Group (SERG), Universiti Teknologi Malaysia
81310 UTM Johor Bahru, Johor, Malaysia**

**[2] Software Engineering Research Group (SERG), Knowledge Economy Research Alliance and Faculty of Computer Science & Information Systems, Universiti Teknologi Malaysia,
81310 UTM Johor Bahru, Johor, Malaysia**

**[3] Soft Computing Research Group (SCRG), Universiti Teknologi Malaysia
81310 UTM Johor Bahru, Johor, Malaysia**

**[4] Software Engineering Research Group (SERG), Universiti Teknologi Malaysia
81310 UTM Johor Bahru, Johor, Malaysia**

Abstract

Online shoppers often have different idea about the same product. They look for the product features that are consistent with their goal. Sometimes a feature might be interesting for one, while it does not make that impression for someone else. Unfortunately, identifying the target product with particular features is a tough task which is not achievable with existing functionality provided by common websites. In this paper, we present a frequent pattern mining algorithm to mine a bunch of reviews and extract product features. Our experimental results indicate that the algorithm outperforms the old pattern mining techniques used by previous researchers.

Keywords: *association rule, pattern mining, product feature, text mining.*

1. Introduction

People usually want to collect more information about a product before purchasing. They usually consider the opinion of other consumers to make decision on their purchase. Nowadays, many websites have been developed which emphasis on participation of users. Some of the websites such as Amazon.com leads people to write their opinion about the products and discuss about the features of that product. It provides a reach information resource on the web. Gathering all these reviews helps manufacturers to aware of the weakness and strength of their product to improve it [6]. But it is not easy to wade through a lot of reviews and read the comments carefully in order to find which attribute or component of the product has received more feedbacks from the consumers. In response, researchers have proposed various techniques to discover such information automatically.

Opinion mining or sentiment analysis aims to determine whether the review sentences deliver a positive, negative or neutral orientation [2]. Product feature extraction is critical to sentiment analysis, because the opinion orientation identification is significantly affected by the target features [13]. Therefore, in this paper we focus on product feature extraction from customer reviews. Specifically, we present an existing pattern mining algorithm to apply for frequent feature extraction from the review sentences. Unlike previous work [10] we only concentrate on those features which have received more opinions from the reviewers.

The rest of this paper is organized as follows. Section 2 discusses the related works on product feature extraction. Then a problem definition is given in section 2. Afterwards, we come up with our method for frequent features identification. Finally, we conclude the study with future works in section 4.

2. Related Works

Existing works [5] [10] [13] [14] which concentrate on unsupervised approach have commonly employed either Information Extraction or Association Rule Mining methods for feature identification.

2.1 Information Extraction

In [5], Popesco proposed OPINE for extracting components and attributes of the products reviewed by the consumers. They compute the pointwise mutual information (PMI) between noun phrases and a set of meronymy discriminators (the semantic relation that holds between a part and the whole) associated with the product class. Their approach is based on the hypothesis that features associated with their product category tend to co-occur in reviews.

2.2 Association Rule Mining

Our work is closely related to Hu and Liu's Work in [10] on extracting product features from reviews. Using association mining they looked for the features that have been talked about by the people frequently. Based on the observation that features are generally nouns or noun phrases, they ran Apriori algorithm on the transaction set of noun/noun phrases to generate frequent itemsets. After producing candidate features they applied compactness pruning and redundancy pruning to remove those features that are not genuine. However, their proposed method was effective in discovering frequent features, but using Apriori leads to increase the execution time while dealing with large databases.
In [13], Chih-Ping extended the above study by adding an additional step to prune possible non-product features and opinion-irrelevant product features. They collect a list of positive and negative words from the general inquirer to determine the subjectivity of a review sentence. Then those frequent features which never or rarely co-occur with any positive or negative adjectives in review sentences are considered as opinion-irrelevant features and removed.
Our work is slightly different from [13] and [10]. We applied different pattern mining algorithm to enhance the precision and performance of the system simultaneously.

3 Problem Definition

This section first defines the general problem of feature identification of reviews and then highlights the specific instance of the problem that we aim to solve. Let us first give the definition of some primary concepts.

Definition 1: *product feature*
Product features refer to all the components, qualities or physical characteristics of a product such as size, color, weight, speed, etc.

Definition 2: *opinion sentence*
An *opinion sentence* is a sentence that consists of at least one product feature and its corresponding opinion word.

Definition 3: *explicit and implicit feature*
An *explicit feature* is a feature of a product which is directly talked about in review sentence. An *implicit feature* is a feature that is not explicitly mentioned in the sentence and it can be implied.

The following sentence shows a negative opinion on a cellphone:

"It is not easy to carry."

"Weight" is an implicit feature of the cellphone which is implied from the sentence.

Definition 4: *frequent and infrequent feature*
A feature *f* is frequent if it appears in majority of the review sentences. *f* is called infrequent if it is only appeared in a few number of reviews.

After putting all these definitions together we go through with general problem of identifying features in the reviews. Most current researches focus on discovering explicit product features. Generally, the current approaches are either supervised or unsupervised. Although, supervised approaches sound to be more accurate, but they need training set that is generated by the human. This approach is effective when the documents are not too away in terms of the subjectivity. This means that if we have two datasets, each of which focuses on a particular topic, the training set for them should be different as well. Let us consider the case we are dealing with opinion orientation of the sentences in a movie and a product review dataset. Normally, opinion words used to express one′s feeling about a movie is different from the situation they are talking about the quality of a product. In a movie dataset some words may carry a negative orientation while the same word in a product review dataset can deliver positive orientation. The same problem may be occurred while dealing with a dataset consisting of reviews on a number of products. Usually, feature words used by the reviewers are varied across different types of product as the components of each product may be unique. So accumulating a set of terms as the training data may bring about running into trouble.

A common unsupervised approach that has proposed by many researchers is based on association mining technique. Focusing on the nouns or noun phrases it is supposed that those nouns that are frequently occurred in the review dataset are most likely to be considered as product features.

In [10], Hu *et al.* used an NLProcesor to parse all the reviews and produce the part-of-speech tag for each word. After identifying nouns they ran an association miner which is based on Apriori algorithm to find frequent itemsets that are likely to be frequent features. This method is simple and efficient and gives reasonable results. However, this technique has some major shortcomings.

Apriori algorithm tests combination of the items without considering of the items ordering. For instance, the words "dvd" and "player" may be occurred in 14 transactions (sentences) as "player dvd" while 87 transactions contain "dvd player". The algorithm cannot recognize the difference between the two situations and it returns only one possible combination such as "palyer dvd" with totally 101 occurrences. However, depending on the chosen threshold, the item "player dvd"'may be considered as an infrequent item and it is not expected to be listed here. Moreover, in case that there exist a large number of frequent patterns, Apriori have to take many scans of large databases and generate huge number of candidates which reduces the performance of the system.

Our work focuses on handling the above problems with the previous work by applying a more efficient frequent pattern mining algorithm.

4 The Proposed Technique

The architectural overview of our feature extraction system is given in Figure 1 and each system component is detailed subsequently.

The system input is a product review dataset including a large number of reviews on products. Reviews in the dataset have been collected and used by [10]. It is a free dataset which is available for download at *http://www.cs.uic.edu/~liub/FBS/CustomerReviewData.zip*. The output of the system will be obtained after passing the following five phases.

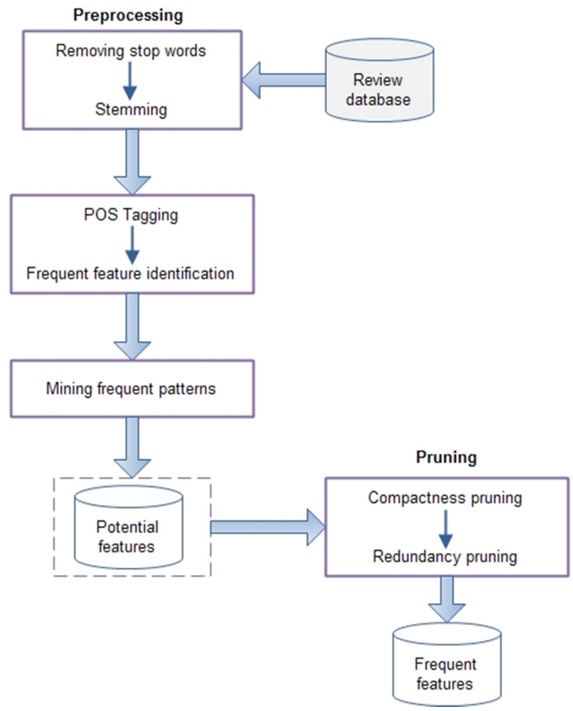

Fig. 1. The System Framework

4.1 Phase 1: Preprocessing

In this work we perform some pre-processing of words including removal of stop words and stemming before going through the next steps.

4.2 Phase 2: Part-of-Speech-Tagging

As the only focused part of the sentences in our work is nouns or noun phrases, we apply a Part-Of-Speech tagger that we developed in PHP to identify the role of the words within the sentences. The following shows a tagged sentence after removing its stop words:

Original sentence: *"The camera is very easy to carry."*
Tagged sentence: *camera/NN easy/JJ carry/VB*

Each sentence is filtered by the identified noun tags and the result is saved in our review dataset.

4.3 Phase 3: Frequent Feature Identification

All the documents in our dataset include sentences that are covering the same topic. In other words, they all created by the customers who are talking about the same objects. Usually, when people discuss and give their opinion on a same thing their words converge. Moreover, a product feature is a noun or noun phrase which is appeared in review sentences. Given the fact, it can be inspired that the nouns with high frequency can most likely be considered as feature words. Frequent pattern mining techniques tend to determine multiple occurrence of the same item. So we have taken the advantage of such techniques in our work in order to find frequent nouns or noun phrases as the potential feature words. Unlike Hu's work, we applied a faster and space-preserving frequent pattern mining algorithm called H-Mine [11] to work with large datasets. Working on the transaction set of nouns or noun phrases coming from previous steps we run H-Mine to find frequent itemsets. The minimum support value of the algorithm is set to 1% meaning that all the patterns that can be found in at least 1% of the review sentences are considered as frequent features.

4.4 Phase 4: Pruning

Association mining algorithms does not consider the position of the items in a given transaction. Thus, after running the algorithm on a sequence of words as an input transaction, it generates a number of candidates that may not be genuine features. On the other hand, in a natural language the words that are appeared together in a specific order usually deliver a particular meaning and they are most likely considered as meaningful phrases. Referring to the above discussion we define a compact feature are a feature phrase that its words do not appear together in the sentence. In this paper we remove non-compact features in the following manner:

```
For each sentence in the review database {
    If (a feature phrase found) {
        For each feature in the sentence {
            Measure the distance between every
            two words;
            If (words distance > 3)
                Remove the feature from the list;
            }
    }
}
```

Suppose that an identified feature is life battery. The algorithm goes through the database and checks if there exist, at least one occurrence of the two words life and battery which appear in a sentence with distance of 3. If it cannot find a sentence, the feature will be removed from the list.
Focusing on features that contain only one word, we also apply another technique to remove redundant features. As a definition, the number of sentences that feature *ftr* is appeared in and there are no superset of *ftr* is called *pure support* of *ftr*. Given the definition, a redundant feature refers to a feature which is subset of another feature phrase and has a pure support lower than minimum p-support. In this work we set the minimum value of p-support into 3 and calculate p-support of every feature . Then those features which have p-support lower than minimum are ignored.

5 Experimental Results and Evaluation

To evaluate the efficiency of our feature extraction system we compare our frequent pattern mining algorithm results with Apriori that has been used by [10]. We first report our experimental results on the performance of H-mine in comparison with Apriori and then evaluate the accuracy of the system while using these two algorithms.

5.1 Performance Measuring

Performance measure is the execution time of the algorithms. To find out the effect of support threshold on the execution time, our system was tested by two different support thresholds. All tests were done on a desktop with an Intel® Core™2 Duo *processor, 4GB Ram,* and a fresh installed *Windows 7* Professional. Figure 1 illustrates average execution times obtained by running Apriori and H-Mine on our review dataset.

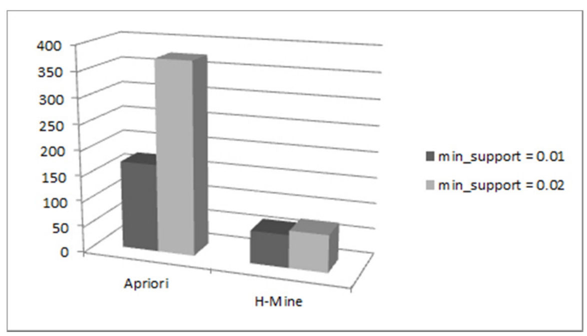

Fig. 2. Average execution time of algorithms

From the figure, we can see that H-mine is much faster than Apriori because of the traversing strategy it follows to mine the dataset. It tries to divide the search space and mine the partitions locally while Apriori follows test-and-generate strategy to mine dataset. Moreover, H-Mine scans database only one time to find the frequent itemsets. Then a tree view of

the data is constructed in the main memory and the algorithm starts to explore the tree in a depth-first search manner.
In another experiment the algorithms were run again after upgrading the support threshold to 0.02. A comparison on the two results reveals that there is an inverse relation between the execution time of the algorithms and the threshold value. The results indicate that increasing the support value leads to decrease the total execution time of the system. It may be explained by the fact that a higher support threshold causes to generate less number of candidates. Accordingly, the total time which is required to perform computations on candidates will reduce.

5.2 Evaluating Accuracy Level

The Accuracy of the system here can be measured by two common metrics, precision and recall. A high precision shows that most of the items returned by the system have been predicted correctly, but there might be some items have not been identified yet. Also, a high recall indicates that less missing items are appeared in the results, but there might be some irrelevant items among them. The best accuracy in this study will be achieved by getting the highest precision and recall simultaneously. On the other hand, the system should predict the maximum number of features correctly while generating less irrelevant results. Hence, another measuring criterion is required to trade off precision versus recall. We use F-Measure as a common measure for testing the accuracy of the system. F-Measure can be interpreted as a weighted average of precision and recall which computes the scores between 0 and 1 to show the worst and the best result respectively.
To evaluate the effect of pruning the results, system was tested both before pruning steps and after pruning.

5.2.1 Pruning Effect

In general, it can be seen in figure 2 and figure 3 that the average of precision was increased on this stage while recall does not show such an improvement. Referring to definition of precision, it can be understood that there is an inverse relation between precision and the number of irrelevant results. Whereas, pruning the results causes to ignore some irrelevant items, the average of precision is increased at this step. Let us say our system has recognized 118 words and phrases as product features. Among these features only 45 items has been predicted correctly. So the precision of the system is obtained from the following formula:

$$\text{Precision} = 45/118=0.38 \qquad (1)$$

If pruning causes to ignore 6 irrelevant items from the list of features identified by the system in the last step then we will have:

$$\text{Precision} = 45/112 = 0.40 \qquad (2)$$

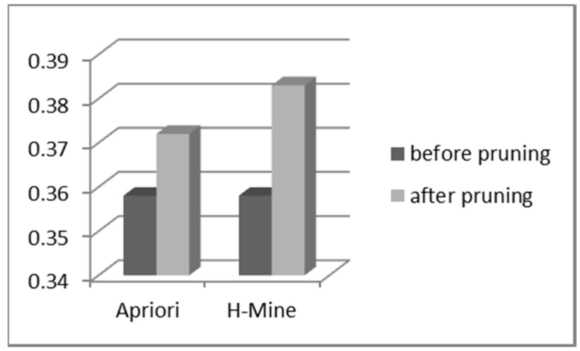

Fig. 3. Average of precision before and after pruning

Compactness pruning and redundancy pruning have a good reduction of incorrectly discovered items. However, a few number of items may be known as either redundant or non-compactness by mistake, and they are removed. Hence, it brings about reducing the recall value while precision is not affected that much.
Let us give an example to elaborate the influence of pruning on the system recall. Imagine that the system has known 83 words as feature. From the discovered features, only 42 words has predicted correctly. If the total number of manual features is 118, the recall of the system will be calculated as follows:

$$Recall = \frac{42}{118} = 0.35 \qquad (3)$$

Compactness pruning and redundancy pruning lead to remove some undesired items. For example, one word may be detected as redundant and 2 words are detected as non-compactness. Therefore, updated value of recall is:

$$Recall_{new} = \frac{39}{118} = 0.33 \qquad (4)$$

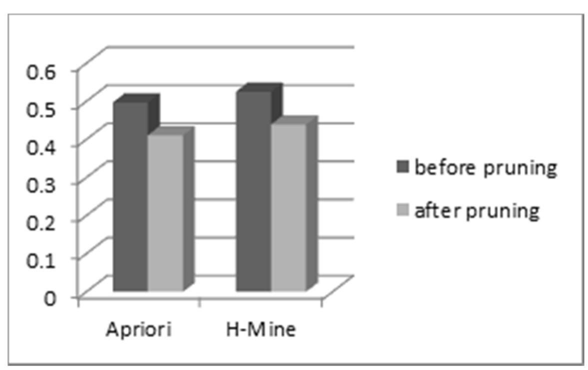

Fig. 4. Average of recall before and after pruning

5.2.2 Trade-off between Precision and Recall

A comparison of the algorithm used by [4] and H-mine that was applied in our work reveals that H-Mine has better precision and recall. Consequently, the highest F-Measure is obtained by H-mine which is illustrated in in figure 4.

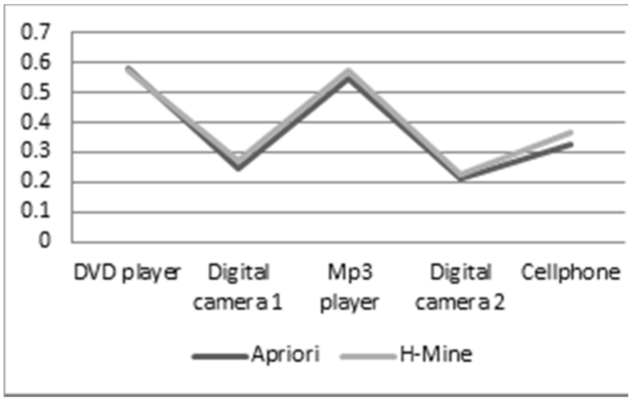

Fig. 5. Calculating f-measure before pruning

Our findings presented in previous section implied that the average of precision was increased while removing redundant and non-compactness features. In addition, it is obtainable from the previous findings that the average of system recall was improved at the same time. Since the speed of reducing the average of recall is faster than precision, it is anticipated that the overall F-Measure is reduced accordingly.

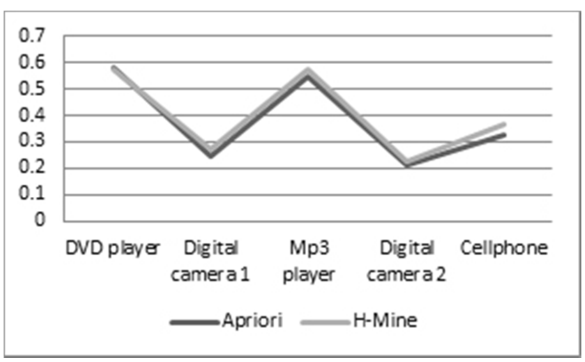

Fig. 6. Calculating f-measure after pruning

6 Conclusions

In this paper, we used a pattern mining algorithm called H-mine to discover features of products from reviews. It is able to deal with two major problems: 1) taking many scans of large databases to generate frequent itemsets, and 2) lack of recognizing transposition of the words while generating new itemsets. In this work we only focused on those features that frequently appear in the review sentences. Our experimental results indicate that our method outperforms the old pattern mining technique used by [4] on both precision and recall.

References

[1] A.-M. Popescu and O. Etzioni, "Extracting Product Features and Opinions from Reviews," in *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, Vancouver, 2005.

[2] B. Liu, "OPINION MINING," In: Encyclopedia of Database Systems, 2004.

[3] C. Agrawal, R. Agrawal and V. Prasad, "Depth first generation of long patterns," in *in Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Boston, 2000.

[4] C. Scaffidi, K. Bierhoff, E. Chang, M. Felker, H. Ng and J. Chun , "Red Opal: Product-Feature Scoring from Reviews," in *Proceedings of the 8th ACM conference on Electronic commerce*, New York, 2007.

[5] G. Carenini, R. T. Ng and E. Zwart, *Extracting Knowledge from Evaluative Text,* 2005.

[6] G. Somprasertsri and P. Lalitrojwong, "Mining Feature-Opinion in Online Customer Reviews for Opinion Summarization," *Journal of Universal Computer Science, vol. 16,* pp. 938-955, 2010.

[7] G. Tianxia, "Processing Sentiments and Opinions in Text: A Survey," 2007.

[8] J. Pei, J. Han, H. Lu, S. Nishio, S. Tang and D. Yang, "H-Mine: Fast and space-pre serving frequent pattern mining in large databases," *IIE TRANSACTIONS,* pp. 593-605, 2007.

[9] J. Yi and W. Niblack, "Sentiment Mining in WebFountain," in *Proceedings of the 21st International Conference on Data Engineering*, 2005.

[10] K. Dave, S. Lawrence and D. M. Pennock, "Mining the peanut gallery: opinion extraction and semantic classification of product reviews," in *Proceedings of the 12th international conference on World Wide*, New York, 2003.

[11] L. Zhang, F. Jing and X. Yan Zhu, "Movie Review Mining and Summarization," Virginia, 2006.

[12] M. Hu and B. Liu, "Mining Opinion Features in Customer Reviews," *American Association for Artificial Intelligence,* pp. 755-760, 2004.

[13] R. Hemalatha, A. Krishnan and R. Hemamathi, "Mining Frequent Item Sets More Efficiently Using ITL Mining," in *3rd International CALIBER*, Ahmedabad, 2005.

[14] W. Chih-Ping, C. Yen-Ming, Y. Chin-Sheng and Y. Christopher C., "Understanding what concerns consumers: a semantic approach to product feature extraction from consumer reviews," in Springer, 2009.

[15] W. Young Kim, J. Suk Ryu, K. I. Kim and U. Mo Kim, "A Method for Opinion Mining of Product Reviews using Association Rules," in Proceedings of the 2nd International Conference on Interaction Sciences: Information Technology, Culture and Human, Seoul, 2009.

[16] X. Ding, B. Liu and P. S. Yu, "A Holistic Lexicon-Based Approach to Opinion Mining," in international conference on Web search and web data mining, California, 2008.

Seyed Hamid Ghorashi is a PhD candidate in computer science at university of technology Malaysia (UTM). His research interests are data mining, text mining, and opinion mining. Hamid received his bachelor in computer hardware from university of Kashan, in 2006 and his master in computer science from university technology of Malaysia in 2012.

Roliana Ibrahim is a senior lecturer at the Faculty of Computer Science and Information Systems, Universiti Teknologi Malaysia. She received her PhD in System Science from Loughborough University and currently a member of Software Engineering Research Group. Her research interests are the adoption of system thinking methodologies and ontology as innovative solutions for complex systems integration and development, data warehousing and data mining.

Shirin Noekhah received her BSc in Software Engineering from Islamic Azad University of Najaf Abad, Iran and MSc in Computer Science from University Technology Malaysia (UTM), Malaysia. Her research interests include Soft Computing and its Applications, Software reliability.

Niloufar Salehi Dastjerdi received her B.Sc. degree in Software Computer Engineering from Azad University of Najafabad, Iran (IAUN) in 2007, and M.Sc. degree in Computer Science from University Technology Malaysia (UTM) in 2012. Her research interests include Natural Language Processing (NLP), Opinion Mining, Text mining and Data Mining.