Mining Opinion Features in Customer Reviews

Minqing Hu and **Bing Liu**

Department of Computer Science
University of Illinois at Chicago
851 South Morgan Street
Chicago, IL 60607-7053
{mhu1, liub}@cs.uic.edu

Abstract

It is a common practice that merchants selling products on the Web ask their customers to review the products and associated services. As e-commerce is becoming more and more popular, the number of customer reviews that a product receives grows rapidly. For a popular product, the number of reviews can be in hundreds. This makes it difficult for a potential customer to read them in order to make a decision on whether to buy the product. In this project, we aim to summarize all the customer reviews of a product. This summarization task is different from traditional text summarization because we are only interested in the specific features of the product that customers have opinions on and also whether the opinions are positive or negative. We do not summarize the reviews by selecting or rewriting a subset of the original sentences from the reviews to capture their main points as in the classic text summarization. In this paper, we only focus on mining opinion/product features that the reviewers have commented on. A number of techniques are presented to mine such features. Our experimental results show that these techniques are highly effective.

Introduction

With the rapid expansion of e-commerce, more and more products are sold on the Web, and more and more people are buying products on the Web. In order to enhance customer satisfaction and their shopping experiences, it has become a common practice for online merchants to enable their customers to review or to express opinions on the products that they buy. With more and more common users becoming comfortable with the Internet, an increasing number of people are writing reviews. As a consequence, the number of reviews that a product receives grows rapidly. Some popular products can get hundreds of reviews at some large merchant sites. This makes it very hard for a potential customer to read them to help him or her to make a decision on whether to buy the product.

In this research, we propose to study the problem of *feature-based opinion summarization* of customer reviews of products sold online. The task is performed in two steps:

1. Identify the features of the product that customers have expressed opinions on (called *opinion features*) and rank the features according to their frequencies that they appear in the reviews.
2. For each feature, we identify how many customer reviews have positive or negative opinions. The specific reviews that express these opinions are attached to the feature. This facilitates browsing of the reviews by potential customers.

We give a simple example to illustrate. Assume that we summarize the reviews of a particular digital camera, *digital_camera_1*. Our summary looks like the following:

```
Digital_camera_1:
  picture quality:
      Positive:  253     <individual reviews>
      Negative:  6       <individual reviews>
  size:
      Positive:  134     <individual reviews>
      Negative:  10      <individual reviews>
  …
```

picture quality and *size* are opinion features. There are 253 customer reviews that express positive opinions about the picture quality, and only 6 that express negative opinions. <individual reviews> points to the specific reviews that give positive (or negative) comments about the feature.

With such a feature-based opinion summary, a potential customer can easily see how the existing customers feel about the digital camera. If he/she is very interested in a particular feature, he/she can drill down by following the <individual reviews> link to see why existing customers like it or what they complain about.

Our task is clearly different from traditional text summarization (Radev and McKeown. 1998; Hovy and Lin 1997) in a number of ways. First of all, our summary is structured rather than another (but shorter) free text document as produced by most text summarization systems. Second, we are only interested in features of the product that customers have opinions on and also whether the opinions are positive or negative. We do not summarize the reviews by selecting or rewriting a subset of the original sentences from the reviews to capture their main points as in traditional text summarization.

In this paper, we only focus on the first step of the review summarization. That is, we aim to mine product features that the reviewers have commented on. The second step of determining whether an opinion is positive or negative will be discussed in a subsequent paper as it is quite involved. Nevertheless, a short introduction of the second step will be given in a later section.

A question that one may ask is “why not ask the merchant or the manufacturer of the product to provide a list of features?” This is a possible approach. However, it has a number of problems: (1) It is hard for a merchant to provide the features because he/she may sell a large number of products. (2) The words used by merchants or the manufacturer may not be the same as those used by common users of the product although they may refer to the same features. This causes problem in identifying what the customers are interested in. Furthermore, customers may comment on the lack of certain features of the product. (3) Customers may comment on some features that the manufacturer has never thought about, i.e., unexpected features. (4) The manufacturer may not want users of its product to know certain weak features.

This paper proposes a number of techniques based on data mining and natural language processing methods to mine opinion/product features. Our experimental results show that these techniques are highly effective.

Related Work

Our work is mainly related to two areas of research, text summarization and terminology identification. The majority of text summarization techniques fall in two categories: template instantiation and text extraction. Work in the former framework includes (DeJong 1982), (Tait 1983), and (Radev and McKeown 1998). They focus on the identification and extraction of certain core entities and facts in a document, which are packaged in a template. This framework requires background analysis to instantiate a template to a suitable level of detail. It is thus not domain independent (Sparck-Jones 1993a, 1993b). Our technique does not fill in any template and is domain independent.

The text extraction framework (Paice 1990; Kupiec, Pedersen, and Chen 1995; Hovy and Lin 1997) identifies some representative sentences to summarize the document. Over the years, many sophisticated techniques were developed, e.g., strong notions of topicality (Hovy and Lin 1997), lexical chains (Barzilay and Elhadad 1997), and discourse structures (Marcu 1997). Our work is different as we do not extract those most representative sentences, but only identify and extract those specific product features and the opinions related with them.

Kan and McKeown (1999) propose a hybrid approach that merges template instantiation with sentence extraction. (Boguraev and Kennedy 1997) also reports a technique that finds a few very prominent expressions, objects or events in a document and use them to help summarize the document. Again, our work is different as we need to find all product features in a set of customer reviews regardless whether they are prominent or not.

Most existing works on text summarization focuses a single document. Some researchers also studied summarization of multiple documents covering similar information. Their main purpose is to summarize the similarities and differences in the information contents of these documents (Mani and Bloedorn 1997). Clearly, our work is related but different.

In terminology identification, there are basically two techniques for discovering terms in corpora: symbolic approaches that rely on syntactic description of terms, namely noun phrases, and statistical approaches that exploiting the fact that the words composing a term tend to be found close to each other and reoccurring (Jacquemin and Bourigault 2001; Justeson and Katz 1995; Daille 1996; Church and Hanks 1990). However, using noun phrases tends to produce too many non-terms, while using reoccurring phrases misses many low frequency terms, terms with variations, and terms with only one word. Our association mining based technique does not have these problems, and we can also find infrequent features by exploiting the fact that we are only interesting in features that the users have expressed opinions on.

Our feature-based opinion summarization system is also related to (Dave, Lawrence and Pennock 2003), in which a semantic classifier of product review sentences is built using a training corpus. However, their system does not mine product features. In addition, our work does not need a training corpus to build a summary.

The Proposed Techniques

Figure 1 gives an architectural overview for our opinion summarization system. The system performs the summarization in two main steps: feature extraction and opinion direction identification. The inputs to the system are a product name and an entry page for all the reviews of the product. The output is the summary of the reviews as the one shown in the introduction section.

Given the inputs, the system first downloads (or crawls) all the reviews, and puts them in the review database. The feature extraction function, which is the focus of this paper, first extracts “hot” features that a lot of people have expressed their opinions on in their reviews, and then finds those infrequent ones. The opinion direction identification function takes the generated features and summarizes the opinions of the feature into 2 categories: positive and negative. In Figure 1, POS tagging is the part-of-speech tagging (Manning and Schütze 1999) from natural language processing. Below, we discuss each of the functions in feature extraction in turn. We will not discuss the final step “Opinion direction identification” as it is not the focus of this paper and it is quite complex and involved (it will be described in a subsequent paper). By direction, we mean whether an opinion is positive or negative.

Part-of-Speech Tagging (POS)

Before discussing the application of part-of-speech tagging from natural language processing, we first give some example sentences from some reviews to describe what kinds of opinions that we will handle.

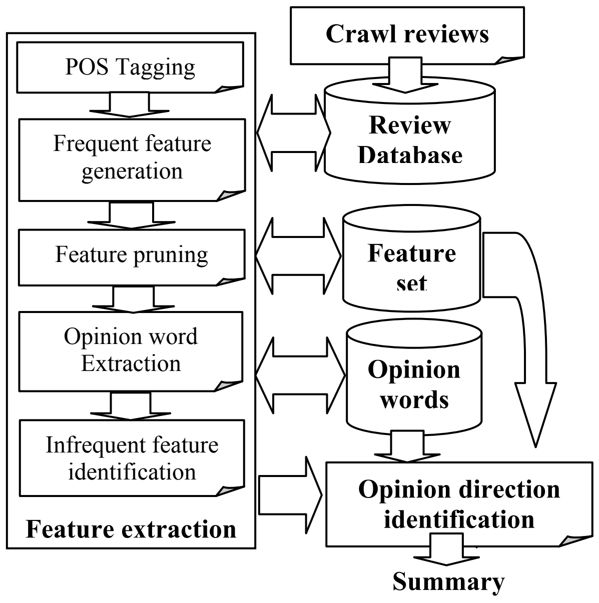

Figure 1: The opinion summarization system

Our system aims to find what people like and dislike about a given product. Therefore how to find out the product features that people talk about is an important step. However, due to the difficulty of natural language understanding, some types of sentences are hard to deal with. Let us see some easy and hard sentences from the reviews of a digital camera:

"The pictures are very clear."

"Overall a fantastic very compact camera."

In the first sentence, the user is satisfied with the picture quality of the camera, *picture* is the feature that the user talks about. Similarly, the second sentence shows that *camera* is the feature that the user expresses his/her opinion. While the features of these two sentences are explicitly mentioned in the sentences, some features are implicit and hard to find. For example,

"While light, it will not easily fit in pockets."

This customer is talking about the *size* of the camera, but the word "size" is not explicitly mentioned in the sentence. To find such implicit features, semantic understanding is needed, which requires more sophisticated techniques. However, implicit features occur much less frequent than explicit ones. Thus in this paper, we focus on finding features that appear explicitly as nouns or noun phrases in the reviews. To identify nouns/noun phrases from the reviews, we use the part-of-speech tagging.

In this work, we use the NLProcessor linguistic parser (NLProcessor 2000), which parses each sentence and yields the part-of-speech tag of each word (whether the word is a noun, verb, adjective, etc) and identifies simple noun and verb groups (syntactic chunking). The following shows a sentence with the POS tags.

<S> <NG><W C='PRP' L='SS' T='w' S='Y'> ***I*** </W> </NG> <VG> <W C='VBP'> ***am*** </W><W C='RB'> ***absolutely*** </W></VG> <W C='IN'> ***in*** </W> <NG> <W C='NN'> ***awe*** </W> </NG> <W C='IN'> ***of*** </W> <NG> <W C='DT'> ***this*** </W> <W C='NN'> ***camera*** </W></NG><W C='.'> **.** </W></S>

The NLProcessor system generates XML output. For instance, <W C='NN'> indicates a noun and <NG> indicates a noun group/noun phrase.

Each sentence is saved in the review database along with the POS tag information of each word in the sentence.

A transaction file is then created for the generation of frequent features in the next step. In this file, each line contains words from a sentence, which includes only pre-processed nouns/noun phrases of the sentence. The reason is that other components of a sentence are unlikely to be product features. Here, pre-processing includes the deletion of stopwords, stemming and fuzzy matching. Fuzzy matching (Jokinen and Ukkonen 1991) is used to deal with word variants or misspellings. For example, "autofocus" and "auto-focus" actually refer to the same feature. All the occurrences of "autofocus" are replaced with "auto-focus".

Frequent Features Generation

This step is to find features that people are most interested in. In order to do this, we use association rule mining (Agrawal and Srikant 1994) to find all frequent itemsets. In our context, an itemset is a set of words or a phrase that occurs together.

Association rule mining is stated as follows: Let $I = \{i_1, \ldots, i_n\}$ be a set of items, and D be a set of transactions (the dataset). Each transaction consists of a subset of items in I. An *association rule* is an implication of the form $X \rightarrow Y$, where $X \subset I$, $Y \subset I$, and $X \cap Y = \emptyset$. The rule $X \rightarrow Y$ holds in D with confidence c if c% of transactions in D that support X also support Y. The rule has support s in D if s% of transactions in D contain $X \cup Y$. The problem of mining association rules is to generate all association rules in D that have support and confidence greater than the user-specified minimum support and minimum confidence.

Mining frequent occurring phrases: Each piece of information extracted above is stored in a dataset called a *transaction* set/file. We then run the association rule miner, CBA (Liu, Hsu and Ma 1998), which is based on the Apriori algorithm in (Agrawal and Srikant 1994). It finds all frequent itemsets in the transaction set. Each resulting frequent itemset is a possible feature. In our work, we define an itemset as frequent if it appears in more than 1% (minimum support) of the review sentences.

The Apriori algorithm works in two steps. In the first step, it finds all *frequent itemsets* from a set of *transactions* that satisfy a user-specified *minimum support*. In the second step, it generates rules from the discovered frequent itemsets. For our task, we only need the first step, i.e., finding frequent itemsets, which are candidate features. In addition, we only need to find frequent itemsets with three words or fewer in this work as we believe that a product feature contains no more than three words (this restriction can be easily relaxed).

The generated frequent itemsets, which are also called candidate *frequent features* in this paper, are stored to the feature set for further processing.

Feature Pruning

Not all frequent features generated by association mining are useful or are genuine features. There are also some uninteresting and redundant ones. Feature pruning aims to remove these incorrect features. We present two types of pruning below.

Compactness pruning: This method checks features that contain at least two words, which we call *feature phrases*, and remove those that are likely to be meaningless.

In association mining, the algorithm does not consider the position of an item (or word) in a transaction (or a sentence). However, in a natural language sentence, words that appear together and in a specific order are more likely to be meaningful phrases. Therefore, some of the frequent feature phrases generated by association mining may not be genuine features. The idea of compactness pruning is to prune those candidate features whose words do not appear together. We use distances among the words in a candidate feature phrase (itemset) to do the pruning.

Definition 1: *compact phrase*

- Let f be a frequent feature phrase and f contains n words. Assume that a sentence s contains f and the sequence of the words in f that appear in s is: $w_1, w_2, ..., w_n$. If the word distance in s between any two adjacent words (w_i and w_{i+1}) in the above sequence is no greater than 3, then we say f is *compact* in s.
- If f occurs in m sentences in the review database, and it is compact in at least 2 of the m sentences, then we call f a *compact feature phrase*.

For example, we have a frequent feature phrase *"digital camera"* and three sentences from the review database contain the phrase:

"I had searched for a digital camera for 3 months."

"This is the best digital camera on the market"

"The camera does not have a digital zoom"

The phrase *digital camera* is compact in the first two sentences but not compact in the last one. However, it is still a compact phrase as it appeared compactly two times.

For a feature phrase and a sentence that contains the phrase, we look at the position information of every word of the phrase and check whether it is compact in the sentence. If we could not find two compact sentences in the review database, we prune the feature phrase.

Redundancy pruning: In this step, we focus on removing redundant features that contain single words. To describe redundant features, we have the following definition:

Definition 2 *p-support (pure support)*

p-support of feature *ftr* is the number of sentences that *ftr* appears in as a noun or noun phrase, and these sentences must contain no feature phrase that is a superset of *ftr*.

p-support is different from the general support in association mining. For example, we have feature *manual*, with the support of 10 sentences. It is a subset of feature phrases *manual mode* and *manual setting* in the review database. Suppose the support of the two feature phrases are 4 and 3 respectively, the two phrases do not appear together in any sentence, and all the features appear as noun/noun phrases. Then the p-support of *manual* would be 3. Recall that we require the feature to appear as a noun or noun phrase as we do not want to find adjectives or adverbs as features.

We use the minimum p-support to prune those redundant features. If a feature has a p-support lower than the minimum p-support (in our system, we set it to 3) and the feature is a subset of another feature phrase (which suggests that the feature alone may not be interesting), it is pruned. For instance, *life* by itself is not a useful feature while *battery life* is a meaningful feature phrase. In the previous example of *manual,* which has a p-support of 3, it is not pruned. This is reasonable considering that *manual* has two senses as noun meaning "references" and adjective meaning "of or relating to hands". Thus all the three features, *manual*, *manual mode*, *manual setting*, could be interesting.

Opinion Words Extraction

Opinion words are words that people use to express a positive or negative opinion. Observing that people often express their opinions of a product feature using opinion words that are located around the feature in the sentence, we can extract opinion words from the review database using all the remaining frequent features (after pruning). For instance, let us look at the following two sentences:

"The strap is horrible and gets in the way of parts of the camera you need access to."

"After nearly 800 pictures I have found that this camera takes incredible pictures."

In the first sentence, *strap*, the feature, is near the opinion word *horrible*. And in the second example, feature *picture* is close to the opinion word *incredible.*

Following from this observation, we can extract opinion words in the following way:

- For each sentence in the review database, if it contains any frequent feature, extract the nearby *adjective*. If such an adjective is found, it is considered an opinion word. A nearby adjective refers to the adjacent adjective that modifies the noun/noun phrase that is a frequent feature.

As shown in the previous example, *horrible* is the adjective that modifies *strap*, and *incredible* is the adjective that modifies *picture*.

We use stemming and fuzzy matching to take care of word variants and misspellings. In this way, we build up an opinion word list, which is used below.

Infrequent Feature Identification

Frequent features are the "hot" features that people are most interested in for a given product. However, there are some features that only a small number of people talked about. These features can also be interesting to some potential customers. The question is how to extract these infrequent features? Considering the following sentences:

"Red eye is very easy to correct."

"The camera comes with an excellent easy to install software"

"The pictures are absolutely amazing"

"The software that comes with it is amazing"

Sentences 1 and 2 share the same opinion word *easy* yet describing different features: sentence 1 is about *red eye*, sentence 2 is about the *software*. Assume that *software* is a frequent feature in our digital camera review database. *red eye* is infrequent but also interesting. Similarly, *amazing* appears in both sentences 3 and 4, but sentence 3 is about *picture* while sentence 4 is about the *software*.

From the examples, we see that people use the same adjective words to describe different subjects. Therefore, we could use the opinion words to look for features that cannot be found in the frequent feature generation step.

In the opinion words generation step, we use frequent features to find adjacent opinion words that modify the features. In this step, we use the known opinion words to find those *nearby* features that opinion words modify. In both steps, we utilize the observation "opinions tend to appear closely together with features". We extract infrequent features using the following procedure:

- For each sentence in the review database, if it contains no frequent feature but one or more opinion words, find the *nearest* noun/noun phrase of the opinion word. The noun/noun phrase is then stored in the feature set as an infrequent feature.

We use the *nearest* noun/noun phrase as the noun/noun phrase that the opinion word modifies because that is what happens most of the time. Since finding the corresponding noun/noun phrases that an opinion word modifies requires natural language understanding, which is difficult with only POS tags, we use this simple heuristic method to find the nearest noun/noun phrase instead. It works quite well.

A problem with the infrequent feature identification using opinion words is that it could also find nouns/noun phrases that are irrelevant to the given product. The reason for this is that people can use common adjectives to describe a lot of subjects, including both interesting features that we want and irrelevant subjects. Considering the following,

"The salesman was easy going and let me try all the models on display."

salesman is not a relevant feature of a product, but it will be found as an infrequent feature because of the nearby opinion word *easy*.

This, however, is not a serious problem since the number of infrequent features, compared with the number of frequent features, is small. They account for around 15-20% of the total number of features as obtained in our experimental results. Infrequent features are generated for completeness. Moreover, frequent features are more important than infrequent ones. Since we rank features according to their p-supports, those wrong infrequent features will be ranked very low and thus will not affect most of the users.

Opinion Sentence orientation determination: After opinion features have been identified, we determine the semantic orientation (i.e., positive or negative) of each opinion sentence. This consists of two steps: (1) for each opinion word in the opinion word list, we identify its semantic orientation using a bootstrapping technique and the WordNet (Miller *et al.* 1990), and (2) we then decide the opinion orientation of each sentence based on the dominant orientation of the opinion words in the sentence. The details are presented in a subsequent paper.

Experiments

We have conducted experiments on the customer reviews of five electronics products: 2 digital cameras, 1 DVD player, 1 mp3 player, and 1 cellular phone. The two websites where we collected the reviews from are Amazon.com and C|net.com. Products in these sites have a large number of reviews. Each of the reviews includes a text review and a title. Additional information available but not used in this project, include date, time, author name

Table 1: Recall and precision at each step of the system

Product name	No. of manual Features	Frequent features (association mining)		Compactness pruning		P-support pruning		Infrequent feature identification	
		Recall	Precision	Recall	Precision	Recall	Precision	Recall	Precision
Digital camera1	79	0.671	0.552	0.658	0.634	0.658	0.825	0.822	0.747
Digital camera2	96	0.594	0.594	0.594	0.679	0.594	0.781	0.792	0.710
Cellular phone	67	0.731	0.563	0.716	0.676	0.716	0.828	0.761	0.718
Mp3 player	57	0.652	0.573	0.652	0.683	0.652	0.754	0.818	0.692
DVD player	49	0.754	0.531	0.754	0.634	0.754	0.765	0.797	0.743
Average	**69**	**0.68**	**0.56**	**0.67**	**0.66**	**0.67**	**0.79**	**0.80**	**0.72**

and location (for Amazon reviews), and ratings.

For each product, we first crawled and downloaded the first 100 reviews. These review documents were then cleaned to remove HTML tags. After that, NLProcessor is used to generate part-of-speech tags. After that, our system is applied to perform feature extraction.

To evaluate the discovered features, a human tagger manually read all the reviews and produced a manual feature list for each product. The features are mostly explicit in opinion sentences, e.g., *pictures* in *"the pictures are absolutely amazing"*. The implicit features such as *size* in "*it fits in a pocket nicely*" are also easy to identify by the human tagger. Column "No. of manual features" in Table 1 shows the number of manual features for each product.

Table 1 gives all the precision and recall results. We evaluated the results at each step of our algorithm. In the table, column 1 lists each product. Columns 3 and 4 give the recall and precision of frequent feature generation for each product, which uses association mining. The results indicate that the frequent features contain a lot of errors. Using this step alone gives poor results, i.e., low precision. Columns 5 and 6 show the corresponding results after compactness pruning is performed. We can see that the precision is improved significantly by this pruning. The recall stays steady. Columns 7 and 8 give the results after pruning using p-support. There is another dramatic improvement in the precision. The recall level has almost no change. The results from Columns 4-8 demonstrate clearly the effectiveness of these two pruning techniques. Columns 9 and 10 give the results after infrequent feature identification is done. The recall is improved dramatically. The precision drops a few percents on average. However, this is not a major problem because the infrequent features are ranked rather low, and thus will not affect most users.

In summary, with the average of recall of 80% and the average precision of 72%, we believe that our techniques are quite promising, and can be used in practical settings.

Conclusion

In this paper, we proposed a number of techniques for mining opinion features from product reviews based on data mining and natural language processing methods. The objective is to produce a feature-based summary of a large number of customer reviews of a product sold online. We believe that this problem will become increasingly important as more people are buying and expressing their opinions on the Web. Our experimental results indicate that the proposed techniques are effective in performing their tasks. In our future work, we plan to further improve these techniques. We also plan to group features according to the strength of the opinions that have been expressed on them, e.g., to determine which features customers strongly like and dislike. This will further improve the feature extraction and the subsequent summarization.

Acknowledgements. This work is supported by the National Science Foundation under the grant IIS-0307239.

References

Agrawal, R. and Srikant, R. 1994. "Fast algorithm for mining association rules." *VLDB'94*, 1994.

Barzilay, R., and Elhadad, M. 1997. Using lexical chains for text summarization. *ACL Workshop on Intelligent, scalable text summarization.*

Boguraev, B., and Kennedy, C. 1997. Salience-based content characterization of text documents. *In Proceedings of the ACL Workshop on Intelligent Scalable Text Summarization.*

Church, K. and Hanks, P. 1990. Word association norms, mutual information and lexicography. *Computational Linguistics*, 16(1) : 22-29.

Daille. 1996. Study and Implementation of Combined Techniques for Automatic Extraction of Terminology. *The Balancing Act: Combining Symbolic and Statistical Approaches to Language Processing*. MIT Press, Cambridge

Dave, K., Lawrence, S., and Pennock, D., 2003. Mining the Peanut Gallery: Opinion Extraction and Semantic Classification of Product Reviews. *WWW-2003.*

DeJong, G. 1982. An Overview of the FRUMP System. *Strategies for Natural Language Parsing*. 149-176

Hovy, E., and Lin, C.Y. 1997. Automated Text Summarization in SUMMARIST. *ACL Workshop on Intelligent, Scalable Text Summarization*

Jacquemin, C., and Bourigault, D. 2001. Term extraction and automatic indexing. In R. Mitkov, editor, *Handbook of Computational Linguistics*. Oxford University Press.

Jokinen P., and Ukkonen, E. 1991. Two algorithms for approximate string matching in static texts. In A. Tarlecki, (ed.), Mathematical Foundations of Computer Science.

Justeson, J. S., and Katz, S.M. 1995. Technical Terminology: some linguistic properties and an algorithm for identification in text. *Natural Language Engineering* 1(1):9-27.

Kan, M. and McKeown, K. 1999. Information Extraction and Summarization: Domain Independence through Focus Types. *Columbia University Technical Report CUCS-030-99.*

Kupiec, J., Pedersen, J., and Chen, F. 1995. A Trainable Document Summarizer. *SIGIR-1995.*

Liu, B., Hsu, W., Ma, Y. 1998. Integrating Classification and Association Rule Mining. *KDD-98*, 1998.

Mani, I., and Bloedorn, E., 1997. Multi-document Summarization by Graph Search and Matching. *AAAI-97*.

Manning, C. and Schütze, H. 1999. *Foundations of Statistical Natural Language Processing*, MIT Press. May 1999.

Marcu, D. 1997. From Discourse Structures to Text Summaries. *ACL Workshop on Intelligent, Scalable Text Summarization*.

Miller, G., Beckwith, R, Fellbaum, C., Gross, D., and Miller, K. 1990. Introduction to WordNet: An on-line lexical database. *International Journal of Lexicography*, 3(4):235-312.

NLProcessor – *Text Analysis Toolkit*. 2000. http://www.infogistics.com/textanalysis.html

Paice, C. D. 1990. Constructing Literature Abstracts by Computer: techniques and prospects. *Information Processing and Management* 26:171-186.

Radev, D. and McKeown, K. 1998. Generating natural language summaries from multiple on-line sources. *Computational Linguistics*, 24(3):469-500, September 1998.

Sparck-Jones, K. 1993a. Discourse Modeling for Automatic Text Summarizing. *Technical Report 290*, University of Cambridge.

Sparck-Jones, K. 1993b. What might be in a summary? *Information Retrieval* 93: 9-26.

Tait, J. 1983. *Automatic Summarizing of English Texts.* Ph.D. Dissertation, University of Cambridge.