Extracting Knowledge from Evaluative Text

Giuseppe Carenini, Raymond T. Ng and Ed Zwart
Computer Science Department
University of British Columbia
2366 Main Mall, Vancouver, B.C. Canada V6T 1Z4
{carenini,rng,ez}@cs.ubc.ca

ABSTRACT
Capturing knowledge from free-form evaluative texts about an entity is a challenging task. New techniques of feature extraction, polarity determination and strength evaluation have been proposed. Feature extraction is particularly important to the task as it provides the underpinnings of the extracted knowledge. The work in this paper introduces an improved method for feature extraction that draws on an existing unsupervised method. By including user-specific prior knowledge of the evaluated entity, we turn the task of feature extraction into one of term similarity by mapping crude (learned) features into a user-defined taxonomy of the entity's features. Results show promise both in terms of the accuracy of the mapping as well as the reduction in the semantic redundancy of crude features.

Categories and Subject Descriptors
I.2.7 [**Artificial Intelligence**]: Natural Language Processing—*Text Analysis*; I.7.5 [**Document and Text Processing**]: Document Capture—*Document Analysis*

General Terms
Experimentation, Measurement

Keywords
feature extraction, multi-document summarization, semantic mapping, term similarity, user input

1. INTRODUCTION
Many corporations and organizations, given the amount of text data they collect, are interested in text mining. But conventional text mining is more successful with structured data than free-form text. Extracting knowledge from free-form text is challenging because of the use of natural language. The solution is to rely on semantic feature extraction offered by NLP techniques. In the past decade most work on extraction has been focused primarily on factual information. Only recent years have witnessed a growing interest in subjective text involving evaluations and affect [1]. The general problem we consider in this paper is how to effectively extract useful information from large corpora of evaluative text.

One important application is the large corpora of customer reviews. The quantity of customer review literature available online is ever-increasing (e.g., consumer electronics). While this literature can be of great strategic value to product designers, planners and manufacturers, the effective processing of this information remains a complex (and expensive) problem. An automated solution has the potential to greatly reduce both the cost and time necessary to keep up with the growing amount of review literature.

While the presentation of this paper focuses on customer reviews, extracting knowledge from evaluative text has many other applications. Commercial applications, such as travel logs, and non-commercial ones, such as office automation tasks (e.g., candidate review and other collaborative processes), would all benefit from automatic summarization of evaluative text. Any entity may be evaluated by numerous reviewers, and therefore justify the need for extracting and summarizing useful information.

Knowledge capture from a large body of text involves two basic tasks. First, it is necessary to extract from the text the most important information. Then such information has to be presented to the user. When the text comprises a large corpus of customer reviews, the information extraction phase can be further specialized as follows. For each review, we need to determine what features of the product are mentioned in the review, how strongly each feature is evaluated and the polarity of the evaluation. For instance, the information extracted from the sentence *"the menus are very easy to navigate but the buttons are somewhat difficult to lo-*

cate" should be that the "menus" and the "buttons" features are evaluated, and that the "menus" receive a very positive evaluation while the "buttons" are evaluated rather negatively. Once this kind of information is extracted from all the reviews, useful knowledge can be generated by aggregating information by features, by polarity and strength. For instance, a useful piece of knowledge extracted from a corpus of reviews could be paraphrased as *"Although most customers really liked the menus, two of them found the menu labels rather confusing"*.

The focus of this paper is on the first information extraction step involved in knowledge capture from customer reviews: determining what features are mentioned in a review. We will not discuss the steps involved in determining polarity and strength any further. We assume this can be done following techniques proposed in [9] for polarity and [18] for strength.

In recent years, it has become clear that for most information extraction tasks manually constructed systems are inadequate, because adapting such systems to domain changes is very expensive and time consuming. Thus, there is a growing interest in using machine learning techniques [6]. Both supervised and unsupervised learning techniques have been applied. Requiring a large annotated corpus, supervised techniques quickly become a less desirable option. Annotated corpora are expensive to create, and the need for a new corpus for each product type makes this approach highly impractical for knowledge capture from product reviews.

On the other hand, unsupervised techniques tend to be more domain independent, and are often much cheaper to develop and modify. One of the most recent unsupervised techniques for extracting features from customer reviews is proposed by Hu and Liu [10]. It is a fully automated system relying on the identification of frequently co-occurring sets of terms. While their approach is successful in terms of precision and recall when tested on an annotated corpus [8], the number of features discovered can be unmanageable. Of the five products in the corpus, four are labeled with over 100 distinct features; one product, an mp3 player, received 188. Furthermore, these feature lists exhibit three additional problems for use as the basis of knowledge capture: (i) they are replete with redundancy, (ii) they contain no hierarchical relationships and (iii) they may not be expressed in a way which is meaningful for the intended user. Redundancy can result either from two semantically identical features expressed in different words such as "remote" and "remote control" or from two highly related terms such as "picture" and "image." A lack of relationship among features appears when two entries, for example "weight" and "size," are not grouped under a parent node such as "physical dimensions." Finally, a naive user may be aware of a digital camera's basic features, such as a "lens" but not some of the more specific or technical ones, such as "optical zoom" or "aperture control." By grouping the latter two in descendant nodes of "lens," the information is much more meaningful to the user than if the three features appeared randomly in a list.

In this paper, we propose a novel approach that addresses all these limitations. As a preview, our approach organizes extracted features in a hierarchical fashion. But instead of building this hierarchy from scratch, it incorporates the user's prior knowledge of the domain features. Specifically, it includes product information in a user-defined taxonomy of features to streamline and better organize the learned features.

The technical contribution to the process is then to employ similarity matching techniques together with the online lexical database WordNet [13] to map learned features into this user-provided taxonomy.

By adding this step into the knowledge capture procedure, we argue that the extracted information will make sense to the user because it is related to her view of the product, cutting down on interpretation time. Furthermore, it will be possible to organize and present the extracted information at different degrees of conciseness by focusing on different levels of the user-provided taxonomy. For instance, a large number of positive comments on a product's menus, buttons and online help may be presented as an aggregate evaluation indicating that customers are reasonably satisfied with the product's user-interface. Finally, since the mapping techniques we propose are not error-proof, we envision a user revision step to allow the user to check and fix which features were not placed or may have been placed incorrectly.

This paper is organized in the following manner. Section 2 contrasts supervised and unsupervised methods of feature learning, and ends with an introduction to our proposed hybrid method. Section 3 outlines the key components of our approach: namely, user-defined features and similarity matching, and introduces several metrics for performing that matching. In Section 4 we present several measures by which to evaluate our approach. A discussion of experimental results comprises Section 5. Section 6 exemplifies the benefits of our approach in terms of captured knowledge when it is applied to a sample corpus. We outline related work in Section 7, and offer concluding remarks as well as our future plans for this work in Section 8.

2. FEATURE EXTRACTION METHODS

In this section we present three options to feature extraction for knowledge capture from product reviews along with their strengths and weaknesses. Figure 1 contains a flow graph of the three options.

2.1 Unsupervised: Method A

The unsupervised method that Hu and Liu present in [10] is a text mining approach to learning product opinion features. It is represented by the left column (Box A) in Figure 1. Their system uses shallow parsing and association rule mining [2] to identify the features discussed in a set of reviews about a given product. The attractiveness of Hu and Liu's system is its simplicity of use. The only input is a set of reviews about a given product. A weakness of this approach, however, is its output. The list of extracted features may be too long and too scattered to provide an adequate basis for effective knowledge extraction. While their system does perform pruning to eliminate simple redundancies, it does not address lexical or term similarity. A second weakness is that features will not be grouped according to their hierarchical relationships to one another.

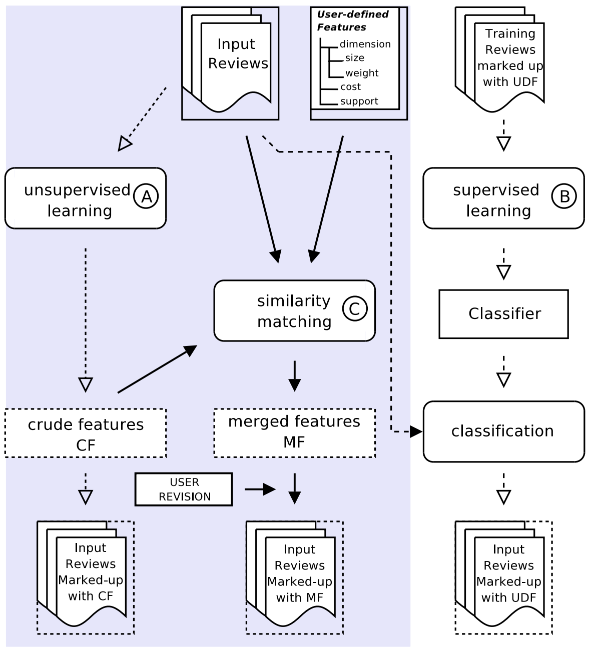

Figure 1: Approaches to feature extraction

2.2 Supervised: Method B

This method, represented by the right column (Box B) in Figure 1, requires a set of training reviews annotated according to user-defined features (hereafter UDF) arranged taxonomically (see Figure 2 for an example). After feeding the annotated reviews into a trainer, the resulting classifier is then applied to the input reviews, which are marked up according to the UDF.

For specific domains, this approach works quite well, but as a general framework for feature extraction, it is impractical mainly because it requires the creation of an annotated corpus of reviews, a costly and time consuming endeavor. One would have to annotate a different corpus for each product type. However, the introduction of the UDF into the process is an interesting one. It captures the important notion that anyone who sets out to capture knowledge from evaluations of a product (or any other entity) is most likely already familiar with its salient features, and that it is desirable to include such user-specific information. Compared to annotating a corpus, the task of creating a UDF taxonomy is considerably less time consuming so as not to be prohibitive, and it can be done by users with varying levels of familiarity with the entity under consideration.

2.3 Our approach: Method C

Our approach, Method C, represented by the middle column (Box C) in Figure 1, borrows from both Methods A and B. From A, we inherit most of its portability. We use Method A's output as input to our system. We attempt to alleviate the unwieldiness of the output of Method A by borrowing prior knowledge and organization from B in the form of the UDF. But we do this without requiring an annotated corpus.

A key component of our approach is the addition of the user-defined taxonomy of features (UDF) to Method A's output, what we refer to as crude features (hereafter CF). We propose to map the output of Method A to the UDF hierarchy thereby eliminating redundancy and providing conceptual organization. Our method is interactive and user guided and tries to strike a balance between supervised and unsupervised approaches.

The second key component of our method is similarity matching. After designing a set of UDF, and running a set of reviews through Method A to produce a set of CF, similarity matching is then performed to map the CF to the UDF. The result is a set of merged features (hereafter MF). This process has several potential benefits. First, redundancy is reduced by grouping similar or identical features under the same UDF. Second, hierarchical relationships between features are introduced and can be exploited in organizing and presenting the extracted information. A third benefit is that such information is framed in a way that the user (or at least the designer of the UDF) envisions the product to be described and reviewed.

Once the MF hierarchy is produced, we envision an interactive user-guided revision process. The user can scan the intermediate output for errors or omissions. First, any misplaced CF can be corrected. Secondly, if the user notices a class of CF features that were inappropriately mapped, she can modify the UDF to accomodate for those features and rerun the similarity matching step. In this way, the UDF is not only reusable, it is adaptable as well. As opposed to the UDF of Method B, where a change would require reannotating the entire corpus, the UDF of our proposed system can be updated on the fly, quickly producing the underpinnings of more accurate and meaningful knowledge.

3. KEY ASPECTS OF OUR APPROACH

3.1 UDF taxonomies

UDF taxonomies are critical components of our approach. In order to reduce as much as possible the influence of our judgments, we started by adopting already existing product taxonomies developed by domain experts, instead of developing our own. We chose the product taxonomies used by Active Sales Assistant^{TM}, a product of Active Decisions, one of the world's leading provider of Guided Selling Solutions, and available at *www.activebuyersguide.com*. We also considered using for the UDF the product description categories employed by the nonprofit organization Consumer Union's ConsumerReports.org®, but ultimately opted for the Active Decisions list because it represents more features and more hierarchical relationships. In our investigation, we started by focusing on two consumer products: digital cameras and DVDs. After a careful inspection of the Active Decisions taxonomy for digital cameras we noticed that it contained a major conceptual inconsistency. The taxonomy was mixing features of the camera with features of the image the camera would generate. For instance, "Viewfinder" and "Optical Zoom" specify camera features, while "Effective Pixels" and "Resolution" pertain more to the image. To address this issue, we split the taxonomy into one for the camera and one for the image. The "Camera" and "Image" taxonomies have three levels each, containing 73 and 13 nodes respectively. Figure 2 shows a small portion of the resulting taxonomies. The DVD taxonomy has only two levels and contains 38 nodes. It did not require any revision.

Figure 2: Partial view of UDF taxonomies for digital camera.

- Camera
  - Lens
    - Aperture Modes
    - Optical Zoom
    - . . .
  - Editing/Viewing
    - Viewfinder
    - . . .
  - Flash
  - . . .

- Image
  - Image Type
    - TIFF
    - JPEG
    - . . .
  - Resolution
    - Effective Pixels
    - Aspect Ratio
  - . . .

3.2 Similarity Matching

The core of our approach to feature extraction for knowledge capture involves matching a flat list of automatically learned features to a taxonomical arrangement provided by the user. Since work in this area is most often referred to as term similarity, we will call product features "terms" hereafter, while maintaining the same shorthand notation (CF and UDF) to distinguish between feature sets. A term is a sequence of one or more words. For a CF term to be matched to a UDF term, some measure of their similarity is required. We measure similarity between two terms by looking at the similarity among the individual words in the terms. In what follows, we introduce three metrics for word similarity and two ways to compute term similarity using the different word metrics. This results in a total of six possible term similarity metrics.

3.2.1 *Word Similarity Metrics*

For all word metrics, v_i and w_j are words within CF and UDF terms, respectively. Words are stemmed, and spelling variants are fixed.

1. Simple string matching is calculated by

   $$str\_match(v_i, w_j) = \begin{cases} 1 & \text{if } v_i \text{ matches } w_j \\ 0 & \text{otherwise} \end{cases}$$

2. *syn_score* employs WordNet and the words' part of speech (POS). In WordNet, lexical items are grouped into synonym sets according to the words' POS and sense. Polysemous words belong to more than one synset. This metric checks whether the two words appear in the same WordNet synset, given their POS. *syns*() returns all synsets a word belongs to for all its senses.

   $$syn\_score(v_i, w_j) = \begin{cases} 1 & \text{if } syns(v_i) \cap syns(w_j) \neq \emptyset \\ 0 & \text{otherwise} \end{cases}$$

3. sim_score_{sm} is actually a class of metrics. It uses one of several similarity measures (sm) described in [3]. This metric also uses WordNet and requires both POS and sense information. The similarity measures are implemented as PERL module WordNet::Similarity [15].

   $$sim\_score_{sm}(v_i, w_j) = \frac{sm(v_i, w_j)}{max(sm)}$$

3.2.2 *Term Similarity Metrics*

For both term metrics, cf_i and udf_j are terms in CF and UDF respectively, while wm stands for any of the word metrics of the previous section (*str_match*, *syn_score* or *sim_score*). v and w are words within cf_i and udf_j.

1. *max* returns the maximum word metric score for $cf_i = \{v_1, ..., v_n\}$ and $udf_j = \{w_1, ..., w_m\}$.

   $$max(cf_i, udf_j) = \max_{i,j}\{wm(v_i, w_j)\}$$

2. *avg* returns the average word metric score for $cf_i = \{v_1, ..., v_n\}$ and $udf_j = \{w_1, ..., w_m\}$.

   $$avg(cf_i, udf_j) = \frac{\frac{\sum_{i=1}^{n} \max_j\{wm(v_i,w_j)\}}{n} + \frac{\sum_{j=1}^{m} \max_i\{wm(v_i,w_j)\}}{m}}{2}$$

3.2.3 Mapping Algorithm

A CF term cf_i is mapped to the UDF term udf_j with which it receives the greatest term similarity metric score (only if greater than some threshold θ). For str_match and syn_score, θ was set to zero. For sim_score, θ has to be set empirically. In the case of tie scores, cf_i is mapped more than once $(udf_j, udf_k, \ldots)$.

4. EVALUATION

To create a gold standard (GS) by which to measure our mapping algorithm's performance, we ran a user study to test a possible mapping of the CF taken from Hu and Liu's corpus of customer reviews [8] to our two UDF taxonomies for digital cameras and DVDs. We showed seven human subjects what we thought to be a good first attempt at a (manual) mapping along with several randomly generated errors, and asked them to make any corrections to the mapping that they deemed appropriate. Based on their input a final version of the GS was created.

The CF terms we use in our experiments are taken from Hu and Liu's annotated corpus [8]. In the digital-camera taxonomy there are 101 CF terms, and 86 UDF terms, while in the DVD taxonomy there are 116 CF terms, and 38 UDF terms. A special pseudo-node for unplaceable CF terms acts as the root in both taxonomies.

For evaluation purposes, we are interested primarily in two assessments: how accurate the mapping algorithm is, i.e., how close its output is to the GS; and to what extent the mapping has reduced redundancy in the CF.

Since we are mapping into a taxonomy, the accuracy of a CF term (cf) is assessed by considering the distance between where it is placed by the mapping algorithm and where it is placed by the GS. The smaller the placement distance is, the more accurate is the mapping. Measuring accuracy in this way reflects how a user might scan results during the user revision process; for instance, a misplacement one edge away is easier to revise than one that is three edges away.

$$placement\_distance(cf_i) = avg(edgeCount(cf_i))$$

The $edgeCount$ returns the number of edges between a placement and the closest correct placement; the average is used for any cf that has been mapped to more than one UDF term. The placement distance of an entire mapping of CF to UDF is the average of each CF term's score.

To measure *reduction in redundancy* in the CF, we subtract the number of UDF terms that received a mapping ($nonEmptyUDF$) from the number of CF terms that were actually mapped ($placedCF$). We divide this number by the total number of CF terms to compute the reduction as a percentage of the CF, namely the percentage of extraneous features in the original flat list of features.

$$redun\_reduc = \frac{|placedCF| - |nonEmptyUDF|}{|CF|}$$

The numerator measures how many CF terms are too similar to be considered as distinct UDF and can therefore be thought of as redundant. For example, if two CF terms such as "picture quality" and "photo quality" are mapped to the same UDF term "Image," one of the two CF terms is redundant. Note that this measure penalizes CF terms that are mapped to multiple UDF terms (by increasing $|nonEmptyUDF|$); simply over-mapping a CF term in hopes of obtaining lower placement distance (better accuracy) will drastically reduce this score.

The GS for digital cameras maps 90 of 101 CF terms to 44 of the 86 UDF terms and leaves 11 unplaced. This means the GS improves the CF terms from the Hu and Liu corpus by 45.5% (i.e., 46 CF terms are redundant). The GS for DVDs maps 64 of 116 CF terms to 14 of the 38 UDF terms and leaves 52 unplaced. This means the GS improves the CF terms from the Hu and Liu corpus by 43.1% (i.e., 50 CF terms are redundant).

5. EXPERIMENTS

Table 1 shows the placement distance and redundancy reduction for three different runs of *str_match*, *syn_score* and two *sim_score* measures *lin* ([12]) and *res* ([16]) using term metric *avg* for the digital camera.

Table 1: Placement distance and redundancy reduction scores for DigCam with term metric *avg*

	First Run		No Repeat		After Revision	
	p_dist	redun	p_dist	redun	p_dist	redun
str_match	.43	.19	.39	.24	.38	.23
syn_score	.45	.21	.40	.28	.39	.27
θ	**sim_score (res)**					
-0.2	.42	.16	.38	.21	.37	.20
-0.4	.42	.23	.36	.28	.36	.27
-0.6	.46	.31	.39	.30	.38	.31
θ	**sim_score (lin)**					
-0.2	.44	.23	.36	.29	.35	.29
-0.4	.43	.31	.38	.33	.39	.33
-0.6	.43	.36	.41	.39	.43	.40

In the first major column (1st Run), we report the results of running the algorithm as described in the previous sections. In the second major column (No Repeat), we show the results of running the algorithm employing the heuristic of not including repeated words in descendant nodes when performing similarity matching. For instance, the UDF "Manual Features" contains four children that repeat the word "manual," resulting in overplacement of CF terms that contain that word, such as “manual function” and “manual mode.” This heuristic avoids the overplacement of the CF terms in all five UDF nodes by allowing the non-repeated, and therefore more meaningful, words to dominate the scoring. The third major column (After Revision) is explained below. In our experiments, we test word similarity measures other than res and lin, but achieve the best results with these. Results for the similarity measures are presented using three different thresholds (θ), calculated as two-, four- and six-tenths of one standard deviation below the average of all term similarity scores.

The lowest placement distance in the first two major columns is .36 edges away from its correct placement. While placement distance scores do not vary greatly among all metrics *str_match*, *syn_score* and *sim_score*, the columns labeled “redun” show that *syn_score* moderately outperforms *str_match* and that the similarity measures significantly outperform the other two measures in reducing the redundancy of the CF. The tradeoff between placement distance and redundancy reduction reveals that as the threshold θ is lowered, more CF terms will be placed into the taxonomy, and more multiple placements will occur. The placement distance score suffers as a result, but not as dramatically as the redundancy reduction score improves.

As mentioned before, our approach offers additional benefits beyond correct placements and reduced redundancy. It provides the user the opportunity not only to revise the mapping, but also to further improve the quality of the UDF. For instance, in examining the results of the digital camera experiment, we found a small number of CF terms that were placed more than three edges away from the correct UDF term. Looking for the reason for these outliers, we detected a small conceptual “error” in the UDF design. A node under the “Image” taxonomy fits better under “Camera.” After fixing the error, the best placement distance score improves to .35 (see third major column of Table 1 – After Revision). The best redundancy reduction score improves to 40%, which corresponds to identifying 40 redundant CF terms (cf. GS identifies 46 CF terms). Taking only a few seconds to attempt, this simple enhancement simulates the same procedure a user might undertake to refine the original taxonomy as a result of seeing particularly odd placements.

Table 2 shows the results for the DVD player, which appear to differ from those for the digital camera. Although it is still true that the similarity measures outperform the other two measures in reducing the redundancy of the CF, for the DVD player we observe a comparable increases in placement distance. This indicates that our finding for the digital camera do not hold in general and that in some domains users may need to tradeoff redundancy reduction and placement distance on an equal basis. Notice that the best redundancy reduction score of .54 equates to identifying 63 redundant CF terms (cf. GS identifies 50). The fact that the algorithm identifies more redundancy than the GS is due to overplacement and this may explain the corresponding increase in placement distance.

Table 2: Placement distance and redundancy reduction scores for DVD player with term metric *avg*

	1st Run		No Repetition	
	p_dist	redun	p_dist	redun
str_match	.31	.19	.27	.21
syn_score	.30	.23	.28	.25
θ	**sim_score (res)**			
-0.2	.39	.30	.39	.32
-0.4	.49	.44	.49	.46
-0.6	.59	.53	.58	.54
θ	**sim_score (lin)**			
-0.2	.42	.36	.40	.38
-0.4	.49	.43	.47	.45
-0.6	.57	.50	.55	.52

In our experiments, we test all combinations of term metrics max and avg with the three word metrics. The scores reported here are for term metric avg only. The results for max were consistently lower, which was expected. avg limits the possibility that a high similarity between two words will dominate the similarity for any two CF and UDF terms.

Notice that these results represent an upper-bound on the performance of Wordnet-based similarity metrics, since in our experiments we assume correct POS and sense tagging for all the CF and UDF.

6. BENEFITS OF CAPTURED KNOWLEDGE: SOME EXAMPLES

Given a large corpus of customer reviews, there are at least two key questions for product designers, planners and manufacturers: what product features are most frequently mentioned by customers? and, do customers dis/agree on their evaluations of such features?

These questions can be answered relying only on the CF, the output of Method A in Figure 1. However, using the merged features (MF) generated by our approach (Method C in Figure 1), additional, more informative versions of these questions can also be answered. For illustration, consider a possible report generated by a system using Method A, when it is applied to the digital camera corpus. As sketched in [10], for each feature the system reports how many times the feature is evaluated in the corpus and how many times the evaluation is positive vs. negative (see Table 3). Features are ordered by frequency (most frequent at the top). To generate this report, we have provided the system with polarity information (which is available in [8]).

Table 3: CF Frequency Statistics

Crude Feature	Total	Pos	Neg
camera	57	55	2
picture	15	13	2
viewfinder	12	1	11
...			
lcd	3	3	0
...			
image quality	1	1	0
image	1	1	0
display	1	1	0
shot	1	1	0

Contrast this with the richer output of a system that by following our approach could rely on the MF [1]. Now in addition to what can be extracted using the CF, the user can also ask the key questions at different level of abstraction and in a terminology she is familiar with. For instance, if the user selects the UDF term "Image" in the digital camera hierarchy (see Figure 2), the output shown in Table 4 is generated.

Table 4: Reduced Redundancy

UDF	Total	Pos	Neg
Image	51	46	5
Image Type	2	1	1
TIFF	1	0	1
...			
Resolution	3	2	1
Effective Pixels	2	2	0
...			

The aggregate values (i.e., Total, Pos and Neg) for each non-leaf feature (e.g., Resolution) are computed by summing up the corresponding values for all the CF terms mapped to the non-leaf feature as well as all those mapped to its descendants in the UDF taxonomy (avoiding double counts due to multiple placements). Similarly, the user might perceive conflicting positive and negative statistics under a non-leaf UDF term such as those shown in Table 5 for the UDF term "Editing/Viewing". Examination of the children nodes reveals that consumers expressed mixed evaluations about "Editing/ Viewing" because they like the "LCD Display", but do not like the "Viewfinder."

Table 5: Informative Mapping Results

UDF	Total	Pos	Neg
Editing/Viewing	17	6	11
LCD Display	4	4	0
Viewfinder	12	1	11

As for redundancy elimination, Table 3 illustrates that a system using only CF generates (and treats as distinct) different entries for "picture," "image quality," "image" and "shot." In contrast, a system using MF would effectively aggregate this information (along with that of other CF terms not shown in Table 3) in a single entry for "Image," as seen in Table 4.

[1]A system using Method B (Figure 1) could also generate a similar output, but remember that Method B requires a UDF annotated corpus.

7. RELATED WORK

Several projects have recently investigated the problem of extracting opinions from customer reviews to support their analysis. As discussed in Section 2, our approach relies on the output of Hu and Liu's system, which identifies the set of crude product features evaluated in a corpus of reviews about the same product. [14] present an unsupervised approach that takes as input a corpus of reviews about several products of the same type and then identifies terms that more specifically characterize the products' similarities and differences in term of customer opinions. However, since these terms do not necessarily correspond to product features, their approach cannot be integrated with ours in a straightforward manner. The same is true for the system presented in [11] which also attempts to extract customer opinions from product reviews without trying to identify the product's features. Furthermore, such a system suffers from the additional limitation of being supervised. It requires a corpus of reviews annotated by the opinion they express overall.

As for our metrics to compute the similarity between two terms, notice that the metrics only consider the similarity between the terms' constituent words. A possible alternative, which has received considerable attention in NLP, would be to follow a corpus-based approach (e.g., [5], [17]). This approach assumes that the meaning of a term is related to how frequently it co-occurs with other terms in text (see the *distributional hypothesis* in [7]). If co-occurrence statistics are collected for each term from a large corpus, the semantic similarity between two terms can then be computed by comparing their respective co-occurrence statistics. For instance, the terms *hash browns* and *French fries* would result to be quite similar because they both co-occur frequently with words like *potato*, *breakfast* and *serve*.

Corpus-based term similarity has been successfully applied to several NLP tasks including query expansion in information retrieval and the automatic construction of thesauri for particular genre or domains. At first glance, it seems that corpus-based term similarity could also effectively support our task of mapping CFs to UDFs. However, we argue, its applicability in this situation is problematic for at least two key reasons. First, it might be extremely difficult to collect robust co-occurrence statistics on abstract (non-leaf) UDF terms. Such features may not be mentioned frequently enough in the corpus, since customers typically refer to more concrete/specific product features in their reviews. As a second reason, we note that corpus-based term similarity requires a large domain specific corpus. But such a corpus may not be available, especially right after the launch of a new product, which is presumably when summaries of customer reviews would be more valuable.

A recent line of research relevant to our investigation is the representation, recognition and generation of paraphrases (i.e. natural language expressions conveying the same information). The most relevant work in this context is [19] which presents techniques to map a given sentence to the corresponding most similar sentence in a set of target sentences. However, these techniques cannot be directly applied to the mapping of CF to UDF because our features are typically much shorter and linguistically simpler than full sentences.

8. CONCLUSION AND FUTURE WORK

In our attempt to find a better solution to capturing knowledge from free-form evaluative text, we argue that the inclusion of user-specific prior knowledge about the evaluated entity is necessary and valuable. New techniques for term similarity combined with relatively straightforward matching metrics provide a powerful and cost-effective way to turn a flat list of automatically extracted features into useful knowledge consisting of a non-redundant set of features which is organized according to a user-defined hierarchy. The introduction of the UDF as well as a method for mapping a flat list of learned features into it are our principal contributions.

Although there are some differences in placement distance among the different word similarity metrics when applied to the two products, they all achieved a remarkably low placement distance of $<= .55$ edge from perfect placement (cf. 2.8 edge error for random placement). Interestingly, the metrics distinguish themselves substantially with respect to the reduction of redundancy score. Redundancy was reduced 40% for the digital camera; 23 features were identified as redundant (cf. 46 in the GS). For the DVD player, 54% of the CF, or 63 terms, were identified as redundant (cf. 50 in the GS). We found that the tradeoff between placement distance and redundancy reduction favours the latter only for the digital camera; for this product placement distance scores increased much more slowly than redundancy reduction scores rose when the threshold for placement was modified. This was not the case however for the DVD player, for which placement distance and redundancy reduction scores increased similarly. This key difference between the two products requires further investigation.

For future work, we plan to introduce more sophisticated NLP techniques to this process, such as including syntactic (headword) information at each UDF node to weight scoring. In our experiments, in order to apply *sim_score* metrics, we manually sense tagged both the CF and UDF. In future experiments, we intend to apply a new word sense disambiguation package (SenseRelate) which looks quite promising. Feature extraction is only the first step in capturing knowledge from evaluative text. For determining polarity and strength, we intend to adapt and extend the techniques proposed in [9] and [18]. To select what knowledge should be presented to the user and decide how to effectively express such knowledge in natural language and/or graphics, we plan to adapt techniques for generating evaluative text presented in [4]. Finally, the approach we have described in this paper is iterative and user-guided. This will require the design of an effective interface to support users in their iterative revision of the CF/UDF mapping and of the UDF taxonomy itself.

9. REFERENCES

[1] AAAI. *Spring Symposium on Exploring Attitude and Effect in Text: Theories and Apps*, 2004.

[2] R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In *Proc. 20th Int'l Conf. VLDB*, 1994.

[3] A. Budanitsky and G. Hirst. Semantic distance in wordnet: An experimental, application-oriented evaluation of five measures. In *Workshop on WordNet and Other Lexical Resources, NAACL*, 2001.

[4] G. Carenini and J. D. Moore. An empirical study of the influence of user tailoring on evaluative argument effectiveness. In *Proc. IJCAI*, 2001.

[5] I. Dagan. *Contextual Word Similarity*, chapter 19, pages 459–476. Marcel Dekker Inc, 2000.

[6] European Conference on AI. *Workshop on ML for Information Extraction*, 2000.

[7] Z. S. Harris. *Mathematical Structures of Language.* Wiley, 1968.

[8] M. Hu and B. Liu. Feature based summary of customer reviews dataset. *http://www.cs.uic.edu/ liub/FBS/FBS.html*, 2004.

[9] M. Hu and B. Liu. Mining and summarizing customer reviews. In *Proc. ACM SIGKDD*, 2004.

[10] M. Hu and B. Liu. Mining opinion features in customer reviews. In *Proc. AAAI*, 2004.

[11] D. Kushal, S. Lawrence, and D. M. Pennock. Mining the peanut gallery: Opinion extraction and semantic classification of product reviews. In *Proc. 12th Int'l Conf. on WWW*, 2003.

[12] D. Lin. An information-theoretic definition of similarity. In *Proc. 2nd Int'l Conf. on ML*, 1998.

[13] G. A. Miller. An online lexical database. *Int'l Journal of Lexicography*, 3(4):235–312, 1990.

[14] S. Morinaga, K. Yamanishi, K. Tateishi, and T. Fukushima. Mining product reputations on the web. In *Proc. ACM SIGKDD*, 2002.

[15] S. Patwardhan and T. Pedersen. Cpan module wordnet::similarity, 2003.

[16] P. Resnik. Using information content to evaluate semantic similarity in a taxonomy. In *IJCAI*, 1995.

[17] J. Weeds, D. Weir, and D. McCarthy. Characterising measures of lexical distributional similarity. In *Proc. COLING*, 2004.

[18] T. Wilson, J. Wiebe, and R. Hwa. Just how mad are you? finding strong and weak opinion clauses. In *Proc. AAAI*, 2004.

[19] I. Zukerman, S. George, and Y. Wen. Lexical paraphrasing for document retrieval and node identification. In *2nd Inter. Workshop on Paraphrasing (IWP2003)*, 2003.