Query-based summarization of customer reviews

Olga Feiguina and Guy Lapalme

RALI
Département d'informatique et de recherche opérationnelle
Université de Montréal
CP 6128, Succ. Centre-Ville
Montréal, Québec, Canada, H3C 3J7
{feiguino,lapalme}@iro.umontreal.ca

Abstract. We describe an architecture for organizing and summarizing consumer reviews about products that have been posted on specialized web sites. The core technology is based on the automatic extraction of product features for which we report experiments on two types of corpora. We thus show that NLP techniques can be fruitfully used in this context for helping consumers sort out the mass of information displayed in such contexts.

1 Introduction

There are now many web sites that gather comments, written by customers, about certain products and services. These web sites are usually maintained either by manufacturers, by sellers (e.g. amazon.com) or by independent people (epinions.com) who (hope to?) make money by selling advertisement space that appears near the comments they display. These sites are quite useful for consumers because they provide *real users'* comments about products. These comments are often summarized by means of global scores or tables of features but the *real* information is still within texts for which there are yet few appropriate processing tools. The users' comments, being written by many different people, are often too numerous, repetitive and unclassified so finding information in this mass of diverse facts and anecdotes is quite an endeavor. As it often happens in our modern world, too much information is worse than not enough, and the time taken by many concerned users to write these comments is thus wasted because few people read and analyze them.

The goal of this paper is to show that it is possible to use NLP technology for organizing and summarizing these comments in order to better help potential buyers. We first describe the architecture of the system based on the automatic identification and merging of product features and then we present the experiments we have carried out and the results we have obtained. More detailed results are described in Feiguina [1].

2 Related Work

Customer reviews have received a lot of attention in the recent years. Research has mainly focused on extracting product features, identifying sentences commenting on them, and classifying reviews or sentences by attitude (positive/negative/neutral). Although our system builds on results obtained in sentiment classification research, we did not work on this aspect of processing customer reviews and focused instead on feature extraction and summarization.

Hu & Liu [2] proposed to summarize customer reviews for a given product by presenting a list of product features with the corresponding counts of positive/negative comments about them. When a user wants more detail, a long list of often repetitive sentences is returned and one of our goals was to solve this problem with our summarization approach. Hu & Liu [2] also worked on feature extraction using an association rule mining algorithm in conjunction with some frequency and overlapping based heuristics to extract the main product features. These features are then used to extract adjacent adjectives which are assumed to be opinion adjectives, which are in turn used to find features that were mentioned only once or several times. As Hu & Liu published their corpus, we could use it to develop some of our ideas.

Also using Hu & Liu's annotated corpus, Popescu and Etzioni [3] worked on feature extraction from reviews within the framework of *KnowItAll* [4]. They calculated point-wise mutual information (PMI) between all noun phrases found in customer reviews and phrases such as `scanner`, `scanner has`, etc. The PMI scores were then fed to a Naive Bayes classifier trained to decide if a given noun phrase is a product feature or not. PMI was calculated using the corpus of reviews as well as using the World Wide Web. The latter gave very good performance: 94% precision, 22% better than Hu and Liu [2], and 77% recall, 3% better than Hu and Liu [2].

These works both relied on a syntactic analysis of the reviews; one of our goals was to limit ourselves to a shallow analysis in order to make the method more language independent. Hu and Liu used a small corpus of reviews while Popescu and Etzioni used the whole Web; our other goal was to strike a middle ground by using an unannotated corpus of reviews which is more *controlled* than the Web but while being able to gather more data than a manually annotated corpus.

Finally, Carenini [5] worked on mapping product features onto an existing taxonomy using various WordNet-based measures of semantic similarity [6]. We used his observations in our experiments on semantic grouping of automatically extracted features that will be presented in section 6.

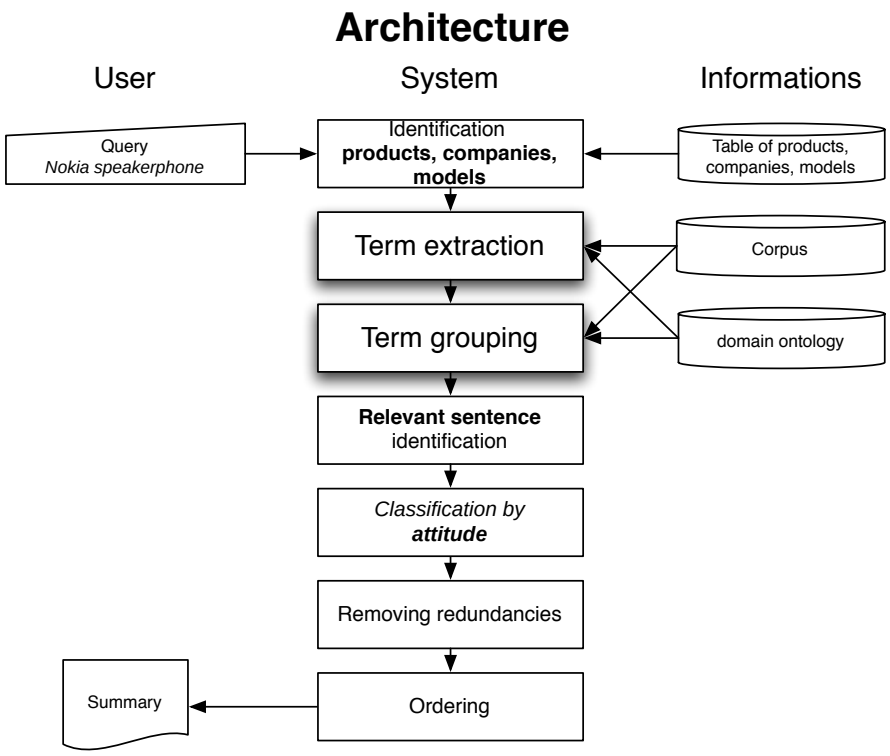

Fig. 1. Architecture of the customer comments summarizing application.

3 Application Architecture

Figure 1 presents the architecture of our application that summarizes collections of customer reviews based on a query. We now give an overview of the modules with a simple example and show the output of our prototype at every stage. The application is invoked once a user enters a query, here `nokia speakerphone`.

Identification of product, company, model Identify, based on the query, what product the user is talking about. In order to do this, we compiled a small database that links company names to products and models of products. Entries of the database are triples of the form {*company*, *product*, *model*} (e.g. {`Canon`, `digital camera`, `Powershot SD300`} for which we allow slight variations such as `Cannon` for `Canon`, etc. Having a pre-compiled list of associated companies, products and models allows for some consistency checks that ensure the user is not mixed up about what a certain company manufactures, what products exist or what product comes in what

models. For our example, we extract `nokia` as the company name and no product or model. What remains of the query is `speakerphone`.

Identification of product features Having identified as much as possible the product, company and model information, we use our lists of features (their creation will be described in Section 5) to identify the ones the user is interested in. For example, given the query `nokia cellphone charger`, we first identify `nokia` as the company, `cellphone` as the product, and then use the list of cellphone features to identify the feature `charger`. In our example, we extract `speakerphone`.

Extraction of relevant sentences We assume that customer reviews are already labeled with the company, product and model they describe. Using the information extracted from the query, we identify the relevant reviews. From them, we extract not only sentences containing features as stated in the query, but also their synonyms. For example, in the electronics domain, for the query `screen`, sentences mentioning `display` are also relevant.

Classification of sentences by attitude The next step is to classify relevant sentences as being either positive or negative. In our prototype, we used a very simple minded approach to this problem: we only label a sentence as positive or negative if the annotation (to be presented in section 4) of a sentence labels it as entirely positive or negative.

Elimination of redundant sentences We define two sentences as redundant if they describe the same features with the same attitude. For example, `the speakerphone works better than any speakerphone i've ever had .` and `this phone has a very cool and useful feature - the speakerphone .` are both positive and talk about the `speakerphone`, so we would consider them redundant. Of course some information is lost through the elimination of sentences, but the essence of the feature being commented on positively is preserved and expressed in natural language.

Often, several features are commented upon in the same sentence. Although the user may not be interested in features not mentioned in the query, we keep the other features. In order to judge if the relevant sentences obtained thus far contain redundant sets, we first extract features other than those mentioned in the query from each sentence. This is done using our precompiled list of features via simple pattern matching. Given two redundant sentences, we keep the longer one, unless one of them has already been judged redundant with another sentence and kept, in which case we keep it again.

Re-ordering of the remaining sentences The remaining sentences form the summary. We group sentences by their attitudes (positive or negative) and we first present sentences that cover more aspects, i.e. more sentences are redundant with those. The result for our example is presented in Figure 2.

Within this overall architecture of a system that summarizes customer reviews based on a query, we mainly focused on the automatic feature extraction that we describe in sections 5 and 6. The next section presents the corpora used in the experiments.

Fig. 2. Excerpts from a summary produced from the query `Nokia speakerphone`.

```
Speakerphone - loud and clear has some nice extra features like currency
converter and a stopwatch. My favorite features, although there are
many, are the speaker phone, the radio and the infrared. The speaker
phone is very functional and I use the phone in the car, very audible
even with freeway noise. [...] The phone book is very user-friendly
and the speakerphone is excellent. The phone has great battery life, fm
radio, excellent signal, hands free speakerphone (which I have to say is
probably my favorite function) and downloadable java apps. [...] Only one
complaint about the speakerphone, you can only activate the speakerphone
feature once the person you are calling answers the phone, not while the
phone is ringing. Great speakerphone and great reception !! recommend!
The speakerphone: People I talk to on the speakerphone are shocked when
the phone comes out at times that I'm even using a speakerphone.
```

4 Corpora and Annotation

In our experiments, we used two corpora, see Table 1.

HL corpus created and annotated by Hu&Liu [2] described in Section 2.
SK corpus was compiled for us by Shahzad Khan (PhD candidate at Cambridge University) who wrote a crawler for `www.epinions.com`. Those reviews were manually sorted by product.

We first relied on the annotation of the HL corpus and we will present results of our experiments using it in the next section. Unfortunately, we noted a series of issues in their annotation that interfered with our approach. Some of them are illustrated by the following examples, all from the first 72 lines of the MP3 player section of the corpus:

- Although implicit features are supposed to be labeled with [u], this is not always the case: `size[-1]##it could be a little bit bigger , but it 's easy to get used to .`
- Sometimes non-features (`looking`) are extracted: `looking[+2]##by the way , it looks nice also .` In this example, `looking` is not in the sentence but is not marked [**u**].
- Features that do not exist are not extracted: `##( i would have appreciated having a firewire plug , however )`. Although future and desired features have a different status, if people comment on them, they need to be on the list of features the system knows about.
- Features are sometimes omitted, so if our extractor found them, they would be counted as errors if we were using this annotation to evaluate it: `##i probably would have liked to have a player in something other than silver / metallic ... like the battery adapters on their usb thumbdrive ( muvo nx ) MP3 player models .` (`battery adapters` not extracted) `##since the front plate is removable to access the battery compartment`

Table 1. The HL and SK corpora: statistics on their size and the description of the annotations in the HL corpus.

Product	HL corpus		SK	
	# reviews	**≈ # words**	**# reviews**	**≈ # words**
Digital camera	79	235	10605	340
Cellphone	41	235	4511	460
MP3 player	95	340	4042	370
DVD player	99	130	3266	280
	314	235	22424	360

Annotation	**Description**
[t]	review titles
xxxx[+|−n]	xxxx is a product feature +|− : positive or negative opinion n strength of the opinion
xxxx[u]	the product feature xxxx is not explicitly mentioned
xxxx[p]	anaphora resolution is necessary to determine the feature

`, aftermarket alternate covers would not be difficult or expensive to make .` (`front plate` not extracted)

Because of these problems, we also compiled our own list of product features. We extracted them from the first 777 sentences of the MP3 section of the HL corpus. Our total was 221 features that we will call the FL (Feiguina and Lapalme) set, as opposed to the HL set extracted from Hu and Liu's annotation. To give a numerical comparison, in the HL set, the total number of features extracted from the whole MP3 section of the corpus (1811 sentences) was only 181.

5 Feature Extraction

The goal of our feature extraction experiments was to extract a set of product features, as complete and noise-free as possible, from a set of customer reviews describing a certain product (cellphones, MP3 players, etc).

5.1 Method

Our method is based on the observation that patterns can be found in the way customers comment on product features. One common pattern is `the` *`feature`* `is` *`adjective`* (e.g. `the speakerphone is great`). This approach is intuitive, requires only a shallow parsing and can make use of large unannotated corpora.

In order to identify useful patterns automatically, we used our annotation of the MP3 section of the HL corpus presented in section 4 and a terminology extractor.

Our terminology extractor, written by Patry and Langlais [7], takes a training text, a set of term examples extracted from this text by a human and a corpus to extract new terms from. It then executes the following steps:

1. Perform part-of-speech tagging of the training text, the training terms and the corpus.
2. Convert the training text to a stream of part-of-speech tags.
3. Learn a language model of terms which represents the most likely part-of-speech sequences within terms.
4. Use the language model for extracting new term candidates from an unseen corpus.
5. Score the terms using AdaBoost on other features such as length, frequency in the corpus, etc.

We used this terminology extractor with the MP3 section of the HL corpus as training text. We used both the features we extracted manually and the HL annotation as training terms. In addition to the term, we used the words around the extracted features.

For example, if from `The silver screen is one of the best screens I've ever had.` we extracted the feature `screen`, we would feed to the terminology extractor `silver screen` or `screen is one` or `silver screen is` as an example of a term depending on the type of context we're considering. The corpus to extract from was the unannotated SK corpus for a given product.

The extracted terms were then *cleaned* from the context. For example, if the context was *WORD TERM WORD* (`silver screen is`), and `clear sound is` were one of the features extracted using the terminology extractor, we would then remove the context to get `sound`.

5.2 Results

In our experiments, we were limited by the need to evaluate manually the terms extracted, so we worked with just three types of context: *TERM WORD WORD* (`screen is one`), *WORD TERM WORD WORD* (`silver screen is one`), and *WORD WORD TERM* (`the silver screen`) and just one product: (Nokia) cellphones.

The extracted terms were evaluated by the first author only, so the scores given in table 2 are approximations of the true precision. For some large sets of terms, we did not evaluate the whole set if the precision stayed the same or got worse as evaluation went on. Our examination of scores produced by the terminology extractor showed that they do not help eliminate the non-features from the list of results.

Our experiments show the good potential value of context for feature identification in customer reviews. We have shown that more elaborate contexts (*word term word word* give good accuracy but very low recall. Of the less restrictive

Table 2. Terminology extraction experiments

Training corpus	context	# extracted terms	estimated precision
HL		12 000	30%
HL	*word term word*	430	55%
HL	*term word word*	810	65%
HL	*word term word word*	53	85%
FL	*word term word*	1200	10%
FL	***term word word***	**800**	**80%**
FL	*word term word word*	26	88%

contexts, *term word word* performed the best; patterns like *term is adj* were very fruitful, like we expected.

Comparing the results gotten using the two different annotations (HL, FL), we see that although in two out of three cases (*term word word* and *word term word word* but not *word term word*) the FL accuracies are higher, the recall is higher for HL in two our of three cases (*word term word word* and *term word word* (marginally) but not *word term word*). The highest score overall score was obtained using FL annotation and the *term word word* context pattern, and the highest accuracy - using FL annotation and the *word term word word* context pattern.

5.3 Future Work

Our experiments show that the approach is promising, but much remains to be explored. In particular, it would be interesting to try different types of context and to study the generality of acquired patterns by verifying if patterns learned from a MP3 player customer reviews can be useful for non-electronics products. Also, it would likely be worthwhile to make the algorithm iterative: using features discovered so far, context patterns including them can be used to take advantage of, for example, sentences where features are listed (e.g. given `known_feature, something, known_feature`, we would say `something` is also a feature). We should also use stricter annotation and evaluation procedures with multiple people so that inter-annotator and evaluator agreement can be measured.

6 Semantic Grouping of Features

Having extracted product features, it is desirable to group them by semantic similarity. For example, features related to the `screen` should be grouped with the ones related to the `display`. Moreover, it would be interesting to identify more distant similarity such as `email` and `messaging`. Feature grouping is especially important given our approach to feature extraction, where features like

color screen and outer screen are extracted: the price for extracting complete features is that there is a multitude of them and many of them are similar.

6.1 Grouping by Salient Words

We carried out two types of preliminary experiments. A first approach was to group extracted features by words they contain, considering only the words salient for the domain. We used TF·IDF (Score(word) = frequency in SK corpus / frequency in Hansard) to determine the salient words, those whose score was above an empirically set threshold. Of the 242 words thus found, 100 were in at least one extracted feature; we will call them 'keywords'. Using keywords to group the features resulted in groups presented in Table 3.

Table 3. Some groupings of features.

email	email, email feature, short emails, email client, writing emails
battery	battery door, battery time, battery power, battery capacity, life battery, battery consumption, battery cover, battery use, battery indicator, rechargeable battery, lithium battery, battery line, battery meter, actual battery, polymer battery, big battery, battery life, battery quality, battery compartment, standard battery, battery level
id	id display, caller id, picture id, id screen, id, id feature
favorite	personal favorite
microphone	external microphone, microphone combo, microphone gain, microphone quality
ericsson	ericsson
charger	extra charger, desktop charger, charger connector, car charger, top charger, charger device, base charger, charger input, travel charger

These groups show that our method allows to group related features such as color screen and outer screen. Our next goal is to establish links between groups based on synonymous words, such as screen and display.

6.2 Grouping by Semantic Similarity

To this end, we attemped to use WordNet-based measures of semantic similarity. We used *path*, *lin*, and *res* similarity measures [6].

As usual when dealing with lexical resources, the first problem to solve was word sense disambiguation. We implemented a simple algorithm for choosing a word sense based on its similarity with a set of domain keywords. From the list of 100 words with high TF·IDF scores that occur in at least one extracted feature, 12 had only one WordNet sense. They were: {keypad, speakerphone, pda (personal digital assistant), phonebook, headset, handset, faceplate,

`voicemail, email, earpiece, messaging, modem`}.

The quality of this WSD algorithm, evaluated using 125 words disambiguated manually, is shown in Table 4.

Table 4. The performance of our word sense disambiguation (WSD) algorithm using three different measures of semantic similarity, evaluated on 125 words with high TF·IDF scores.

Sim measure	WSD precision
path	51%
res	47%
lin	55%

Using this WSD module, we used the following algorithm after having computed the semantic similarity of all feature pairs (if a feature consists of multiple words, an average over the semantic similarity of all pairs of words is taken).

- If two keywords have semantic similarity greater than an empirically determined threshold T, we merge the feature groups of these keywords.
- If a non-grouped feature has semantic similarity greater than T with a keyword, add it to that keyword's group.
- If a feature has semantic similarity greater than T with another grouped feature, add it to all of that feature's groups.
- If two non-grouped features have semantic similarity greater than T, create a new group with those features in it.

We iterate over these steps until no further changes are made to maximize the grouping. The resulting grouping steps were evaluated manually by the first author. The best performance of about 67% was observed when the measure of similarity *path* was used; of the 55 proposed groupings, 27 seemed good to us. Some examples of the groupings we judged good/bad are as follows:

- Good groupings:
  - `switch`, `button` (*two keywords*)
  - `interior panel`, `inside panel` (*two ungrouped features*)
  - `net access`, `internet` (*an ungrouped feature and a keyword*)
  - `message system`, `voice message` (*an ungrouped feature and a grouped feature*)
- Bad groupings:
  - `device`, `alarm` (*two keywords*)
  - `list`, `menu` (*an ungrouped feature and a keyword*)
  - `input system`, `message system` (*an ungrouped feature and a grouped feature*)

Although the good groupings look promising, there is a number of necessary improvements to eliminate the bad ones. Many bad groupings are the result of hyponyms and hypronyms getting high semantic similarity scores (e.g. `device` and `alarm`). This could be handled by creating a special measure of semantic similarity where such relationships between words receive a lower score. Alternatively, a clustering algorithm more sophisticated than our threshold-based one should be investigated. Overall, although the groupings generated by our algorithm are not enough for automatic ontology construction, they offer a good starting point to a human ontology engineer.

7 Future Work

An interesting extension to add to our architecture would be a *cooperative analysis* such as the one proposed by Benamara and St-Dizier [8] to ensure that the initial query is coherent, i.e. that the product or the features really relate to the right manufacturer. This would imply some consistency checks with a database of products with their set of features and their manufacturer. This type of database is already present for other aspects of these types of customer sites.

We have done experiments only with electronics related products for which our patterns seemed to apply, but it would be interesting to apply them to comments on other types of products or on bigger corpora to see how such an approach would scale up.

8 Conclusion

We have shown in this paper how to organize consumers' comments based on an automatic term extraction mechanism and a grouping around issues that were deemed relevant in these comments. Of course, it remains to be seen if such a type of organization is really useful for consumers. To our knowledge, this work is the first attempt to produce natural language summaries for this type of texts. We have shown that although these free comments are written by non specialists, they seem to use similar text pattern that allowed us to identify the mains points described by these texts, to regroup similar texts while removing the ones that merely repeat already reported information.

Acknowledgment

We would like to thank Shahzad Khan for allowing us to use his web crawler. We thank also Balázs Kégl for discussions and comments about machine learning algorithms. We also thank the members of the RALI laboratory in particular Philippe Langlais and Alexandre Patry who developed the term extractor we used in our experiments and Fabrizio Gotti for his help on many technical and scientific matters. This work has been partly funded by a NSERC discovery grant.

References

1. Feiguina, O.: Automatic summarization of customer reviews. Master's thesis, University of Montreal (2006)
2. Hu, M., Liu, B.: Mining and summarizing customer reviews. In: Proceedings of the 2004 ACM SIGKDD international conference on Knowledge discovery and data mining (KDD 2004), New York, NY, USA, ACM Press (2004) 168–177
3. Popescu, A.M., Etzioni, O.: Extracting product features and opinions from reviews. In: Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing, Vancouver, British Columbia, Canada, Association for Computational Linguistics (October 2005) 339–346
4. Etzioni, O., Cafarella, M., Downey, D., Popescu, A.M., Shaked, T., Soderland, S., Weld, D.S., Yates, A.: Unsupervised named-entity extraction from the web: an experimental study. Artificial Intelligence **165**(1) (2005) 91–134
5. Carenini, G., Ng, R.T., Zwart, E.: Extracting knowledge from evaluative text. In: Proceedings of the 3rd international conference on Knowledge capture (K-CAP 2005), New York, NY, USA, ACM Press (2005) 11–18
6. Patwardhan, S., Banerjee, S., Pedersen, T.: Using measures of semantic relatedness for word sense disambiguation. In: Proceedings of the Fourth International Conference on Intelligent Text Processing and Computational Linguistics, Mexico City, Mexico (2003) 241–257
7. Patry, A., Langlais, P.: Corpus-based terminology extraction. In: Proceedings of the 7th International Conference on Terminology and Knowledge Engineering, Copenhagen, Denmark (August 2005) 313–321
8. Benamara, F., Saint-Dizier, P.: Construction de réponses coopératives : du corpus à la modélisation informatique. In: Revue Québéquoise de linguistique. Number 3 in 18. (septembre 2004) 34–59