Opinion classification Through information extraction

L. Dini[1], G. Mazzini[1]
[1]CELI,
Turin, Italy

Abstract

In this paper we present an application of information extraction technologies to data mining. The system has the goal of producing reliable reports on opinions that customers express about companies and/or products. Information extraction is regarded as an *enabling technology* by which structured databases can be populated from unstructured texts available on the web. The approach can be considered as an alternative to standard text mining techniques: rather than applying data mining algorithms on textual inputs, we propose to apply syntactic and semantic processing in order to disclose structured information which abstracts completely from linear order of words and language dependent constructions.

1 Introduction

1.1 Information sources for customer opinion discovery

As CRM is becoming more and more a factor of success for many companies, we observe a more and more urgent need of keeping track of *indirect customer opinions*, i.e. those opinions which are not addressed directly to the company but to some "third party" institution. For instance, there are sites which collect opinions of customers about certain products and services, possibly providing legal advices. Complaints and appraisals are usually archived and made available to other customers (see for instance the site of the Italian Association for the Rights of Customers (www.aduc.it) or the commercial site www.complaints.com).

Even more prominently, users tend to "freely associate" into opinion sharing communities such as newsgroups, forums, chat communities etc. The set of documents stored into these information containers is a valuable source of information on the status of companies and their products. For instance the brand *Nokia* was cited into 104 out of 452 messages randomly downloaded from the newsgroup *it.tcl.cellulari* (notice that there are two different newsgroups specialized for *Nokia* mobile phones, i.e. *it.tcl.cellulari.nokia and it.tcl.hardware.nokia*)

The landscape becomes even more impressive if we consider the web as a whole. For instance, if we type the word "Gilera" (Italian Motorbike brand) on a popular search engine, we will obtain about 100.000 hits. The same search on the set of available newsgroups produces about 30.000 hits. Overall these 130.000 documents, after the exclusion of false matches, homonymies and institutional web sites, provide a clear snapshot about the reputation of the company and its products.

If data about customers opinions could be collected and stored into appropriate database structures, they would represent an ideal data source for the application of data mining algorithms ([1]). For this opportunity to become actual, however, a further step must be performed: textual sources (possibly in several different languages) must be transformed into pure information, i.e. ultimately into database records. This is what the present paper is about.

1.2 Traditional approaches

The need of automating the process of data capturing finalized to the application of data mining techniques is evident from the consideration of the amount of data available on the web. In the case of Gilera, we have to face 100.000 hits plus 30.000 newsgroup quotes. If we assume an average time of ten minutes needed by a human operator to consult a page and formalize the opinions/facts contained therein, it emerges that the "scanning time" for such a brand amounts to roughly 1350 person days, i.e. 67 person months. In these cases, random sampling is only a partial solution in that it presupposes partition of input data along meaningful dimensions. This partition, however involve in turns either the application of automatic classification techniques or human scanning of the "rough" input base.

In order to face problems of this nature two different branches of Data Mining were developed, i.e. web mining and text mining. The former is virtually of no help in order to tackle cases such as the one proposed here, as it presupposes in general the existence of "stereotyped" web pages to which wrappers are applied (with different degree of flexibility) in order to extract relevant information ([2], [3]): it is clear that the output of search engines are web pages whose internal structure is virtually unpredictable, ranging from dynamic output of online database to nearly plain text documents (such as newsgroup archives).

Concerning text mining, most traditional methods are statistical in nature, and relates either on patterns identification or on concept induction via word occurrence. In the former case (e.g. [4]) identification of customer opinions is quite difficult, as the appraisal/criticism patterns are very varied and can hardly be recovered by pure character matching techniques. As for the latter (cf. [5] for a survey), it is clear that "word bags" techniques can be useful in identifying the topic of a page (for instance, its relevance to customer opinion tasks) but they cannot go as far as identifying the polarity of the judgement, which often depends on words with low discrimination power (stop words, e.g. *not*).

What we need are techniques which go far behind the conception of the text of a sequence of character or as a bag of words. We need a system able to understand text.

1.3 The information extraction approach

In the first part of this paper we will describe an information extraction system customized in order to satisfy needs arising from data mining applied to customer opinion discovery. We will show how intelligent crawling techniques can be coupled together with shallow syntactic analysis and semantic finite state transducers in order to capture customer opinions.

In the second part, we will go one step further and we will provide a roadmap of how pioneering research will be able to go beyond simple identification of a positive/negative judgement and will provide the means to obtain a finer grained classification of customer judgements both in terms of granularity (opinion expressed as integers rather than booleans) and specificity (which are the possible causes for a certain judgement?). Such an approach, which presupposes tight integration of shallow parsing approaches with deep processing performed by cognitively based grammars, is currently under experimentation in the context of the Deep Thought project (IST-2001-27836).

2 Information extraction for opinion discovery tasks

In this section we will describe the basic workflow of an information extraction system tailored to extract customer opinions from newsgroups and web pages. The basic components of the system will be described in detail, with a particular enphasis on configuration tasks, a typical bottleneck in the development of information extraction applications. All the described components are currently integrated into an opinion discovery application through which CELI is offering services of database population to its customers. Overall, we can guarantee that the application performs textual analysis with a recall of 52% and a precision of 92 % (the figures have been computed according to the method proposed in [6]).

2.1 Sources of Information

The first thing to set up into an “information discovery task” such as the one described here, is the identification of the information sources. So far we mentioned two basic sources, i.e. search engines and newsgroups. However, given the average amount of results returned by queries about brands and/or products, some information filtering strategy need to be set up. Indeed if the information extraction system had to process and compute information from all relevant pages, processing time might become a serious bottleneck for the whole mining process. Moreover, there are certain kinds of ambiguities which are better discerned by an information retrieval engine than by an information extraction system. This is for instance the case of ambiguous brands and product names, where the following ambiguity patterns can be detected:

- Company name-Company name: *Mitchell* is the company name of both a fishing rods producing company and a software vendor.
- Product name – Product Name: *Excalibur* is, among other things, both a computer game and a LaTeX spell checker.
- Company name – common word: *Oracle* is both the name of a company and a common English word (on newsgroups, extraction engines cannot rely on capitalization, which is often misused).
- Product name – common word: *Panda* is both the name of a car by *Fiat* and the name of an animal in many languages.

In all these cases the addition of some discriminating words (relating, for instance, to the domain/field to which a certain product belongs) to the search engine queries can discriminate with a reasonable precision relevant from irrelevant hits. In the case of newsgroups, as messages are usually classified into rather fine grained topic categories, we can safely assume that a correct category selection is enough to guarantee message relevance.

2.2 Crawling and Filtering

From a purely conceptual point of view the crawling component adopted in opinion discovery tasks does not need to be particularly sophisticated. Indeed the basic actions it has to perform are the following:

- For search engines:
  - issue relevant queries to one or more search engine;
  - download the hits;
  - cycle through the *next* button to reach the remaining hits.
- For newsgroup servers:
  - navigate through the newsgroup hierarchy;
  - cycle through the available threads using the *next* link;
  - cycle through all messages available in a thread.

Such a goal can be easily attained by most commercial crawling engines. For the experiment described in this paper we used *InfoBroker* a script based crawling engine developed by CELI in the context of the MIETTA-II project (IST-2000-30161). Such a crawler has the additional advantage of being able recover from critical situations which might arise in precessing intensive tasks. For instance, in case of net crash or system crash it is able to restart without having to redo previously done downloading work. Also it is able to cross check result hits from multiple queries in order to avoid the download of redundant documents.

Once all the relevant documents are downloaded, the phase of format conversion can start, with the goal of transforming web pages into plain text documents to be passed as input to the information extraction engine. For standard web pages, this conversion amounts to stripping HTML tags, while preserving only “visible” text, i.e. text which could be read by a human operator browsing that page. For emails and newsgroup items this process is complicated by the following factors:

- Emails contain header information (such as date, sender's email, etc.) which might be relevant for data mining procedures and which must be consequently recognized and stored.
- The interpretation of newsgroup items is often dependent on the *context*, i.e. on the list of messages to which the current message is an answer. As previous messages are often reported in the email using standard graphical conventions (most often a ">" sign) it is extremely important 1) to avoid extracting information from reported emails; 2) to preserve the text of reported email for the phase of semantic interpretation, in order to provide a correct resolution of anaphoric references (see below).

Consequently for documents which belongs to the category "newsgroup item" a special conversion filter is applied with the goal of marking reported messages in a format which can be easily recognized by the extractor.

2.3 The extraction process

In its "modern" technological incarnation IE can be seen as an information transformation chain from plain text segments to structured database records. The basic idea is to pipeline several processing components in order to obtain more and more usable results. While different systems can vary a lot in architectural terms (also depending on the language to be analysed), we can usually identify at least three different big components:

- A *tokenization* component, with the role of identifying words and possibly categorizing them according to features which cannot be enumerated in a lexicon. Typically a tokenization component will recognize person names and positions, company names, product names, addresses, several types of codes, etc.
- A *lexical access component*, with the role of associating categories to words. The associated categories might be either plain syntactic categories or semantic categories, or a mixture of the two, depending of the goal of the application.
- A *finite state processing tool* with the role of analysing the linear order of words in order to identify useful sequences. The finite states processing phase might aim at performing syntactic recognition (with different levels of granularity) or it might try to identify directly the constructs which are relevant for the application, with little attention for underlying syntactic structures. Usually several finite tool grammars are pipe-lined in order to obtain stepwise refinement of the information.

The information extraction system we adopted (*Sophia 2.1*, cf. [11]) comes with an already configured set of linguistic "cartridges" (i.e. set of lexical and syntactic rules) for Italian, English and French. These are important resources as they can shorten by a lot configuration times, which is one of the traditional bottleneck of standard information extraction systems. In particular, for the opinion discovery task we had to perform virtually no modification to the tokenization rules and to the lexicons (except for the addition of lists of terminological words). As for finite state rules, we must make a distinction between syntactic level rules and semantic level rules.

Syntactic analysis in information extraction systems is usually limited to *chunking*. Chunking is a shallow processing theory introduced by [7]. Its main goal consists in the identification of groups of words which forms linguistic unit with one selected word which is semantically more prominent than the others (*head*). For instance, the sentence *I do have some problems with my Gilera to start in the morning* can receive the following syntactic interpretation (square brackets represents grouping and underlined words represents heads):

(1) `[I][do have][some problems][with my Gilera][to start][in the morning]`

Chunking rules are quite similar across different domains and stylistic registers, so the base rule set provided with *Sophia 2.1* did not undergo any change in order to fit the needs of the opinion discovery application. The only modification to the syntactic processing component was represented by the introduction of "compounding rules". This rules are aimed at identifying groups of words which "behave" as a single word, in the sense that they designate a single concept (*2 stroke oil*, *header pipes*). As compound words vary from domain to domain, corpus based techniques have been used to speed up the acquisition process of compound words which most frequently occurred in the document base.

Semantic configuration represented the major effort to adapt the generic information extraction system to the task of customer opinion discovery. In *Sophia 2.1* semantic configuration is achieved by editing rules which describe a linguistic context on their left hand side and an annotated structure (comparable to an attribute value matrix) on their right hand side. For instance:

```
 *{ANY-POSITIVE}[sem=HAVE] *{ANY} [sem=PROBLEM var=v1]
 [sem=PRODUCT, syn=WITH var=v2] *{ANY var=v3}[syn=fullstop]
                                  ⇓
[ polarity='negative' expression=v1 entity=v2 context=v3 ]
```

Basically, this rule will match a sequence of chunks composed of a verb with the same semantic category as the verb *have*, a noun belonging to the semantic category `PROBLEM` and an expression denoting a product name (`sem=PRODUCT`) introduced by the preposition *with*. The semantic category is the same as the head of the chunk, whereas the sytactic category is assigned directly in chunking rules. {ANY} is a macro matching most of available linguistic chunks, excluding punctuation, while {ANY-POSITIVE} is a macro matching the same chunks as {ANY}, with the exclusion of negative polarity items (e.g. *do not*). `v1`, `v2` and `v3` are variables which are referred to in the right hand side of the rule. When the left hand side matches the input sequence of chunks, the right hand side structure is produced, where variables are replaced by the value of the head of the corresponding chunks. We call such an output a *filled template*. Given an input such as the one in example (1), the application of the rule will produce a filled template like

```
>Can anybody tell me what the Gilera 180 is like.
>Any problems, comments, or anything relating to buying a Gilera.
>Cheers,
>Hitman
I've had the Runner 180 since May & have clocked up about 6,900
miles so far. I use the bike everyday for work (a round trip of about
45 miles) and enjoy every minute.
```

Figure 1: A sample newsgroup message

the following one:

```
[polarity='negative' expression='problems' entity='Gilera'
context='start morning']
```

The grammar for opinion discovery contains 43 semantic rules which apply to the input text and produce a sequence of filled templates. Of course, the degree of informativeness of templates might vary from text to text: for instance product name, problem statements and context of the problem might be distributed across several structures. The process of gluing together this information in order to produce a meaningful representation is a well known challenge in IE: it involves the treatment of complex phenomena such as anaphora resolution, context based inferences, temporal reasoning etc. *Sophia2.1* offers two different mechanism in order to merge information stemming from different filled templates. We distinguish between problems which are mostly dependent on inferences (e.g. temporal inference) and problems which are mostly dependent on the order of templates (e.g. pronominal resolutions). In the former case, the resulting filled templates are transformed into propositional logic clauses and passed to a prolog-like engine to perform the desired inferences. In the latter, they are converted into an order preserving XML document, and passed to a set of XSL stylesheets, which have been empowered to perform language based operation through the standard XSL-extension mechanism.

After the application of these template merging components, a newsgroup message such as the one in figure 1 can be reduced to a semantic representation like the following:

```
[polarity='positive' expression='enjoy' entity='Gilera 180'
context='every minute']
```

Semantic configuration and template filling configuration are quite time consuming operations. However, for opinion discovery tasks, such an operation must be performed only once: in order to gather opinion about different products or brand no reconfiguration is needed, only minor adjustments connected with different domains (lexicon and compounding). Also, the time needed to configure the system is dramatically reduced by adopting the *Sophia 2.1* workbench which provides an integrated development environment, partially inspired to common software engineering IDE such as *JBuilder*^{TM}. In particular the workbench for the language engineer provides the following tools:

- **Specialized editors:** each linguistic component comes with a specialized editor, with possibility of rule search, pretty printing, fontification, parenthesis matching and syntax checking.
- **Result browsing:** A special browser is used for displaying results of analysis. It allows both browsing inside the linguistic structure of analysed sentences and queries over rules and syntactic and semantic categories.
- **Corpus analysis:** Configured extraction grammars can be run over a corpus to verify the effects of recent modifications. A special module performs statistical analysis to verify deviances with respect to previous runs.
- **Project management:** each set of grammars is conceptually grouped into a *project*. The user can create how many projects she wants in order to manage different applications. Grammars and all linguistic components can be shared across projects or can originate independent development lines (much like *branches* in the *CVS* program).

2.3.1 Database population

Ultimately, the information extraction engine will deliver a set of filled templates encoded as XML documents. These templates are transformed on the fly into SQL insert queries to populate the database on which data mining procedures will be applied. Abstracting from the physical database structure (which may vary from application to application and which, in most cases, must fit a pre-existing corporate organization) the range of information which the system is currently able to extract is described in table 1.

3 Mixed HPSG Approach

The information extraction approach presented in the previous section offers the possibility of disclosing important customer opinion information to data mining algorithms. However, more effective knowledge discovery procedures could be applied if information of the kind that can be collected by forms or phone interview would be available. This amounts to tailor the information extraction system in such a way that it can produce finer grained representations. For instance it could be very useful to replace the yes/no judgement for product/brand appraisal with a fuzzier judgement, which expresses customer satisfaction on a scale from 0 to 10. Also it would be crucial to have a more in depth comprehension of the causes for a certain judgement. For instance in cases of mechanical faults it would be important to know which parts broke, in order to identify weak points in the production chain (cf. [8]). Or, in the case of complaints linked to lack of assistance it would be crucial to know the geographical location of the customer.

These kinds of upgrades are not outside the expressive power of an information extraction systems. However, given the specificity of the data to be extracted, in order to achieve such a level of refinement a *specific* semantic configuration would

Table 1: Types of information.

Field	Possible value	Comment
entity	Any string	The name of the product or company.
polarity	positive/negative	Whether the judgement is positive or not.
qualifi cation	fault/assistance/ documentation/price	Possible causes for a negative judgement.
expression	Any string	The words used to express the opinion.
context	Any string	Linguistic context in which the judgement was given
email	email	The email of the person judging the product (if available)
website	URL	The URL of the site where the document was found.

be needed for each different application domain. This trivially follows from the fact that the way customers have to express faults in a mechanical piece of a motor bike might be very different from the one they would adopt for talking about a fault in a mobile phone.

One significative step towards the minimization of the configuration overload, would be represented by a system able to produce *full* semantic representations of the input texts: rather than transforming natural language sentences into structure which are meaningful just for the specific task, an easy configurable system should map them to domain independent semantic representations. Structures like the one in figure 2, besides being formally more tractable than arbitrary templates, can be much more easily processed by sets of domain dependent rules. Indeed domain specific computation should only take into account general functional concepts rather than word order and syntactic peculiarities. *Full* semantic representations

$$\begin{bmatrix} \text{pred} & \text{have} \\ \text{subj} & \text{me} \\ \text{obj} & \begin{bmatrix} \text{pred} & \text{problem} \\ \text{quant} & \text{some} \\ \text{about} & \text{Gilera} \\ \text{restriction} & \begin{bmatrix} \text{pred} & \text{start} \\ \text{when} & \text{morning} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

Figure 2: An example of a full semantic representation.

of natural language texts can be obtained by *deep* computational grammars, i.e. grammars which parse each input sentence and “navigate” the resulting syntactic structure in order to obtain well formed semantic representations. This grammars are usually “cognitively ground” in the sense that the backbone of the rules they apply is derived from theoretically sound theories of language and language use.

On the purely computational side, however, deep grammatical organizations present at least two problems:

- Given the complexity of human language and its combinatorial features, processing times are often unacceptable for real applications.
- They rely on a “neat” conception of language usage, which might sometime diverge from actual usage as found in real word texts. If a fragment of language does not fit the adopted system of rules, parsing stops and no semantic representation at all is retrieved for the input sentence.

One of the major goals of the project Deep Thought (IST-2001-27836), is to take advantage of both information extraction shallow techniques and deep grammatical organization by integrating them with the goal of producing a full text analysis *under all circumstances* and at *reasonable computational costs*. Of course not any grammatical theory could fit the constraints imposed by such an integration. In the context of the project HPSG (*Head Driven Phrase Structure Grammar*, [9]) has been selected as the reference grammatical theory. The reasons for such a choice are linked to the modular organization of HPSG, which is based on general “universal” linguistic principles that are tailored (more specifically “subtyped”) for different languages and different stylistical registers. Other advantages offered by such a theory are the following ones:

- its computational model is formally tractable and has already received a number of implementation.
- recent works on stochastic method and disambiguation techniques largely improved the computational efficiency of implemented models.
- The kind of semantic representation produced by most of the available HPSG grammars (Minimal Recursion Semantics , cf. [10]) represents an optimal trade off between completeness and simplicity.

The information extraction system and the HPSG parser should integrate their capacity of processing in the context of a multilayered architecture, where data are exchanged as XML documents obeying a predefined interchange format. In particular, the major interactions bewteen the two systems should be oriented towards the following lines:

- The information extraction system will work as a pre-processor for the HPSG parser. It will identify relevant passages where customer’s opinion is expressed and pass them sentence by sentence to the HPSG parser.
- If the HPSG parser fails to parse a sentence, this is sent back to the IE engine: the IE engine will then apply traditional finite state automata techniques in order to derive a structure as similar as possible to the one which should be delivered by the HPSG parser. As an alternative process, the IE engine might decide to further segment the sentence into phrases and have the single phrases parsed by the HPSG grammar.
- In any case at the end a bag of minimal recursion semantics representations will be produced. Such a bag will be again transmitted to the IE engine in order to distill the required domain dependent templates.

4 Conclusions

In this paper we presented information extraction as an enabling technology aimed at providing structured input for the application of data mining algorithms. The choice of customer opinions discovery as an application domain is motivated both by undeniable market needs and by the presence of quantitatively relevant textual inputs. We are now planning to expand the proposed methodology to other important domains, most prominently to automatic market analisys (both in terms of competitors and potential customers) and knowledge discovery tasks in the biomedical domain.

References

[1] Tan, P.-N., Blau, H., Harp, S. & Goldman, R., Textual Data Mining of Service Center Call Records. *Proc. of the 6th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, eds. R. Ramakrishnan, S. Stolfo, R. Bayardo & I. Parsa, Boston, MA, pp. 417–423, 2000.

[2] Liu, L., Pu, C., Han, W., XWRAP: An XML-Enabled Wrapper Construction System for Web Information Sources. *Int, Conf. on Data Engineering (ICDE)*, pp. 611–621, 2000.

[3] Sahuguet, A., Azavant, F., Building light-weight wrappers for legacy web datasources using w4f. *Int. Conf. on Very Large Databases (VLDB)*), pp. 38–741, 1999.

[4] Fujino, R., Arimura, H.,& Arikawa, S., Discovering Unordered and Ordered Phrase Association Patterns for Text Mining. *Pacific-Asia Conference on Knowledge Discovery and Data Mining*,pp. 281–293, 2000.

[5] Kobayashi,M. & Takeda,K., Information retrieval on the web. *ACM Computing Surveys*, **32(2)**, pp. 144–173, 2000.

[6] Grisham, R. Information Extraction: Techniques and Challenges.*Information Extraction*, ed. M.T. Pazienza, Springer-Verlag, Berlin Heidelberg,pp. 10–27, 1997.

[7] Abney, S.P.. Parsing by Chunks. *Principle-Based Parsing: Computation and Psycholinguistics*, eds. Berwick, R.C., Abney, S.P. & and Tenny,C., Kluwer Academic Publishers, Boston, editors, pp. 257–278, 1991.

[8] Yairi, T., Kato, Y. & Hori, K., Fault Detection by Mining Association Rules from House-keeping Data, *Proc. of Int. Symp. on Artificial Intelligence, Robotics and Automation in Space*, i-SAIRAS. 2001.

[9] Pollard, C. & Sag, I., *Head-Driven Phrase Structure Grammar*, CSLI Publications, Stanford, 1994.

[10] Copestake, A., Flickinger, D., Malouf, R., Riehemann,S ., & Sag, I., Translation using Minimal Recursion Semantics. *Proc. of the 6th Int. Conf. on Theoretical and Methodological Issues in Machine Translation*,Leuven, 1995.

[11] Dini, L. & Di Tomaso, V., Corpus Linguistics for Application Development, *International Journal of Corpus Linguistics*,**3(2)**, 1998.