Deriving Marketing Intelligence from Online Discussion

Natalie Glance
nglance@intelliseek.com

Matthew Hurst
mhurst@intelliseek.com

Kamal Nigam
knigam@intelliseek.com

Matthew Siegler
msiegler@intelliseek.com

Robert Stockton
rstockton@intelliseek.com

Takashi Tomokiyo
ttomokiyo@intelliseek.com

Intelliseek Applied Research Center
Pittsburgh, PA 15217

ABSTRACT

Weblogs and message boards provide online forums for discussion that record the voice of the public. Woven into this mass of discussion is a wide range of opinion and commentary about consumer products. This presents an opportunity for companies to understand and respond to the consumer by analyzing this unsolicited feedback. Given the volume, format and content of the data, the appropriate approach to understand this data is to use large-scale web and text data mining technologies.

This paper argues that applications for mining large volumes of textual data for marketing intelligence should provide two key elements: a suite of powerful mining and visualization technologies and an interactive analysis environment which allows for rapid generation and testing of hypotheses. This paper presents such a system that gathers and annotates online discussion relating to consumer products using a wide variety of state-of-the-art techniques, including crawling, wrapping, search, text classification and computational linguistics. Marketing intelligence is derived through an interactive analysis framework uniquely configured to leverage the connectivity and content of annotated online discussion.

Categories and Subject Descriptors: H.3.3: Information Search and Retrieval

General Terms: Algorithms, Experimentation

Keywords: text mining, content systems, computational linguistics, machine learning, information retrieval

1. INTRODUCTION

The Internet has enabled many online forms of conversation and communication, such as e-mail, chat groups, newsgroups, message boards, and, more recently, weblogs. Some channels are private, some public, some mixed. In many areas, there is a wealth of consumer information to be tapped from online public communications. For example, there are message boards devoted to a specific gaming platform, newsgroups centered around a particular make and model of motorcycle, and weblogs devoted to a new drug on the market. Both the consumer and the corporation can benefit if online consumer sentiment is attended to: the consumer has a voice to which the corporation can respond, both on the personal level and on the product design level.

This paper describes an end-to-end commercial system that is used to support a number of marketing intelligence and business intelligence applications. In short, we describe a mature system which leverages online data to help make informed and timely decisions with respect to brands, products and strategies in the corporate space. This system processes online content for entities interested in tracking the opinion of the online public (often as a proxy for the general public). The applications that this data is put to range from:

- Early alerting - informing subscribers when a rare but critical, or even fatal, condition occurs.
- Buzz tracking - following trends in topics of discussion and understanding what new topics are forming.
- Sentiment mining - extracting aggregate measures of positive vs. negative opinion.

Early implementations of these applications in the industry were enabled by sample-and-analyze systems where a human analyst read a tiny fraction of the data available and made observations and recommendations. As these approaches can not handle realistically-sized data sets, modern approaches are built on technology solutions which use comprehensive crawling, text mining, classification and other data driven methods to describe the opinion reported in online data.

Other systems described in research literature have also focused on aggregating knowledge from the web. The WebKB project [9] was an early effort to automatically extract factual information about computer science research departments, people, and research projects using departmental web sites. Their emphasis was on the application of machine learning techniques to the extraction of data and facts, without emphasis placed on the access or understanding of the data. The CiteSeer project [5] extracts information from

KDD'05, August 21–24, 2005, Chicago, Illinois, USA.

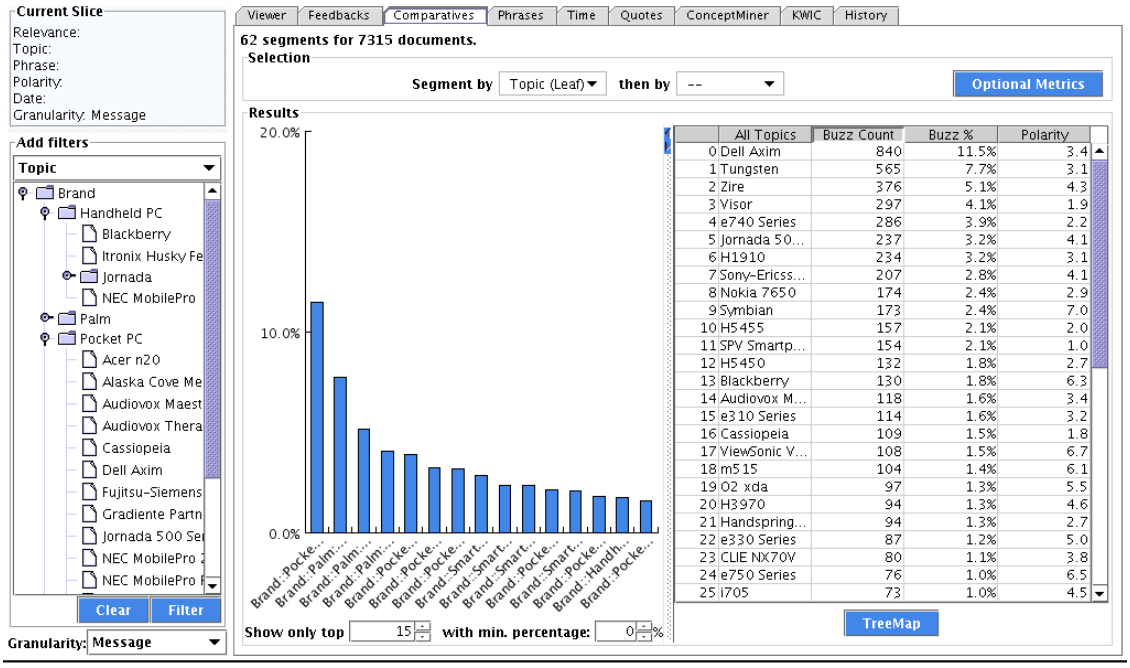

Figure 1: A breakdown of selected messages by brand, along with several metrics. The Buzz Count metric measures overall volume of discussion, where the Polarity metric measures overall sentiment towards the brand indicated.

online research papers. This project emphasizes the collection and extraction of information, as well as making the data publicly accessible through a search interface. Our system goes beyond collection, extraction, and access, and also provides a significant capability to interactively analyze the data to form an understanding of the aggregate expression of knowledge backed by the data. Our work is similar to the Takumi project [20] in providing a system that does analysis over extracted information from text data—call center data in their case. Our work emphasizes challenges created by focusing on web data, and the appropriate technologies used to meet these challenges.

The application described here applies, develops and contributes to many areas of research. The requirements of the application have directed specific research in the areas of focused crawling and wrapping, active learning, sentiment analysis, phrase discovery, and aggregate metrics. Bringing these technologies together in an application constrained by document type, genre, and language allows us to leverage the promise of text mining for the domain of consumer sentiment analysis.

2. CASE STUDY

This section presents a specific example of how a project can be used to discover marketing intelligence from internet discussion data. As is described in the following sections, a project is configured to collect internet discussion in a target domain, classify the discussion across a number of domain-specific topics (e.g. brand, feature, price) and perform a base analysis of the sentiment regarding combinations of topics. A typical project will analyze anywhere from tens of thousands of messages to tens of millions of messages.

The following case study presents such a project in the domain of handheld computers, including PDAs, Pocket PCs, and Smartphones. Some of the basic questions a brand manager might ask are "What are people saying about my brand?" and "What do people like and dislike about my brand?". This paper argues that these questions are best answered through interactive analysis of the data. Manual review of a small fraction of the data or simple search and IR techniques over the whole data are generally not comprehensive or deep enough to quickly provide answers to these basic questions.

Figure 1 shows a screenshot of our interactive analysis tool. One way to analyze messages is through a top-down methodology, that starts with broad aggregate findings about a brand, and then follows through to understand the drivers of those findings. The Comparatives analysis shown is a simple way of breaking down the messages and generating a variety of metrics over each segment. Figure 1 shows all the messages about handhelds broken down by the brand being discussed. The Dell Axim is the most "popular" brand, as measured by buzz volume, capturing 12% of all discussion about handheld devices. However, by a measure of overall sentiment, the Dell Axim does not do so well. The Polarity column shows a 1-10 score representing the aggregate measure of sentiment about this brand (see Section 4.2). The Axim's score of 3.4 is a relatively low score. As a brand manager, you would like to drill down on these high-volume but low-sentiment aggregate measures to understand the drivers of this discussion.

With a few clicks in the application, an analyst can select just the messages saying negative things about the Axim. By analyzing these messages, one can understand drivers of the Polarity metric value. The Phrases tab identifies the distinguishing words and phrases for negative Axim discussion

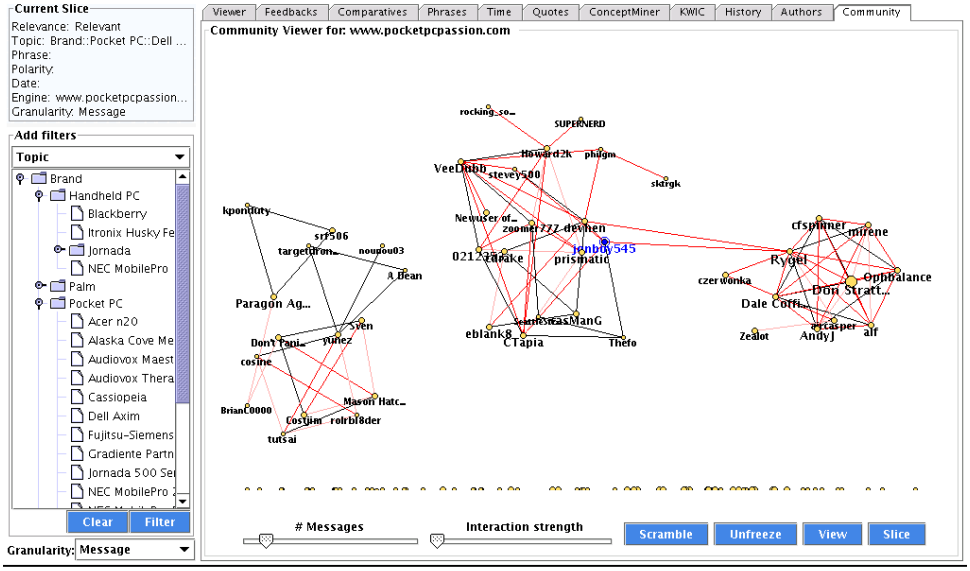

Figure 2: A display of the social network analysis for discussion about the Dell Axim on a single message board. The messages are dominated by three separate discussions. Drilling down on the right-most cluster reveals a discussion complaining about the poor quality of the sound hardware and IR ports on the Axim.

Keywords	Keyphrases
Axim	Dell Axim
X5	Pocket PC
Dell	my Dell Axim
par	Dell Axim X5
today	battery life
ROM	SD card
problem	Toshiba e740
incompatible	CF slot

Table 1: The top eight words and phrases for negative comments about the Dell Axim. Words like "ROM", "incompatible" and phrases like "SD card" and "CF slot" are at the top of the list, indicating specific problems people have with the Dell Axim.

through a combination of statistical and NLP techniques. Table 1 shows the top eight words and phrases, as calculated by our phrase-finding technology described in Section 4.1. Further drilling down on these words and phrases to the messages containing them reveals, for example, that a number of "SD cards" are "incompatible" with the Axim, and that "ROM" updates are needed to make Personal Internet Explorer work correctly on the Axim.

A second way of analyzing data is through a bottom-up methodology. Here, analysis starts with all relevant discussion to identify nuggets or clusters of information that can be distilled down through interactive analysis into actionable intelligence. One such technique in our application is a social network analysis. Figure 2 displays the social network for discussion regarding the Dell Axim on one of the popular Pocket PC discussion boards. Each node in the graph is an author, and links between authors are created when authors interact by posting in the same thread. The length of each link connecting two nodes is inversely proportional to the strength of their interaction, determined through the

- It is very sad that the Axim's audio AND Irda output are so sub-par, because it is otherwise a great Pocket PC
- Long story made short: the Axim has a considerably inferior audio output than any other Pocket PC we have ever tested.
- When we tested it we found that there was a problem with the audio output of the Axim.
- The Dell Axim has a lousy IR transmitter AND a lousy headphone jack.
- I would hate to tell you this is going to help you out, since the performance of the Axim audio output is spotty at best.

Table 2: Five representative automatically extracted negative sentences about the Dell Axim within a cluster of discussion identified by social network analysis. The quotes indicate that the main topic of discussion within the group was the poor quality of the Axim's audio and IR components.

frequency of participation in the same threads. Each author is weighted (and displayed by font size) by their authority, as determined by their propensity to spark discussions with many interactions. Authors and links can be filtered through threshold selection sliders to allow the analyst to focus in on just the most salient clusters of discussion. Figure 2 shows three such clusters. By selecting just the right-most cluster of messages, the analyst can quickly proceed to the Quotes analysis, which displays sentences with high sentiment about the selected brand (see Section 3.6). Table 2

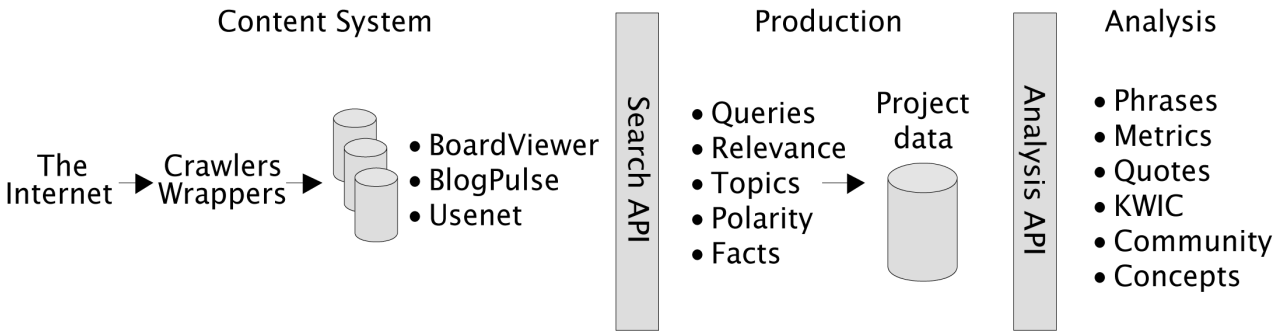

Figure 3: Overview of the system showing content collection, production and analysis.

shows results of this analysis for negative sentiment about the Axim within that authorial cluster. The quotes clearly show a brand manager that a group of people are unhappy about the audio and IR components of the Dell Axim.

This case study illustrates two key points. First, an interactive analysis system can be used to quickly derive marketing intelligence from large amounts of online discussion. Second, the integration of many different state-of-the-art technologies are necessary to enable such a system. The remainder of this paper describes the technologies underlying the different components of the system.

3. SYSTEM OVERVIEW

The case study above illustrates the power of interactive analytics over data collected from online sources. These analytics represent an application of a large-scale web enabled system.

The system is comprised of three main components, as shown in Figure 3. The content system crawls the web for weblog, message board and Usenet content and populates internal search indices (as described in Section 3.1). The production system uses a set of queries to retrieve messages from the content stores and applies analyses to the messages, producing a set of tagged messages. These tagged messages form the project data over which interactive analytics are run using the application shown in the previous section.

3.1 Content System

Discovery and harvesting of message data is the first component of our system. We have modules for harvesting from Usenet newsgroups, message boards and weblogs. Discovery and harvest from Usenet newsgroups is straightforward since the Usenet distribution mechanism makes discovery of newsgroups simple and because Usenet posts are well-defined structures.

On the other hand, both message boards and weblogs pose both discovery and harvesting difficulties. Discovery entails finding message boards and weblogs pertinent to a particular domain. Harvesting consists of extracting message board posts and weblog posts from semi-structured web pages. The first step in both cases is designing a crawling strategy. The second step entails a kind of reverse engineering for message boards and weblogs to reconstruct message board posts and weblog posts. The solutions devised depend on the data source. Below, we discuss first our approach to crawling and segmenting weblogs and second our approach to crawling and segmenting message boards.

3.1.1 Weblogs

Weblogging has emerged in the past few years as a new grassroots publishing medium. Like electronic mail and the web itself, weblogging has taken off. Recent estimates place the number of active weblogs at over 4 million and doubling in size every 5 months[1].

The weblogging microcosm has evolved into a distinct form, into a community of publishers. The strong sense of community amongst bloggers distinguishes weblogs from the various forms of online publications such as online journals, 'zines and newsletters that flourished in the early days of the web and from traditional media such as newspapers, magazines and television. The use of weblogs primarily for publishing, as opposed to discussion, differentiates blogs from other online community forums, such as newsgroups and message boards. Often referred to as the blogosphere, the network of bloggers is a thriving ecosystem, with its own internally driven dynamics.

More recently, marketing groups are becoming aware of the strong influence that highly networked bloggers can have over their readers. The top tier of bloggers have as many readers as regional newspaper columnists. Even more interesting to marketers is the middle segment of bloggers who have managed to carve out audiences of hundreds to thousands of readers with specific interests.

There is no comprehensive centralized directory of weblogs. In fact, an opt-in directory would become stale very quickly as the half-life of a weblog is approximately four months[2]. However, one key aspect of weblog authoring software is that it automatically pings one or more centralized services when the weblog is updated. (In some cases, this feature can be turned off or customized.) We collect the list of recently updated weblogs from these services. These services include the update lists from: blogrolling.com, weblogs.com, diaryland.com, livejournal.com, xanga.com, blo.gs and myspace.com. From this list of updated weblogs, we can retrieve the weblog page itself. As of 11/2004, we are finding about 300,000 updated weblogs per day.

Our goal is to harvest newly published weblog *posts* from the updated weblogs. Thus, we have the task of extracting structured data from the semi-structured weblog home page: the title, date, author, permalink and content of each newly published post. We call this task *weblog segmentation*. We use a model-based approach to segment weblogs into posts. We assume that the format of a weblog is:

[1]http://www.sifry.com/alerts/archives/000387.html
[2]http://www.perseus.com/blogsurvey/

- weblog: (entry) (entry)+
- entry: date (post)+
- post: [title] content

The *title* field is optional, and we require that there be at least two entries on the weblog home page.

The first step in segmentation is to recognize the dates in the weblog. This is done using a date extractor. We then sort the dates into groups with equivalent xpaths. Next, we apply a set of heuristics to choose which group corresponds to the dates of the entries for the weblog. For example, the list of dates must be monotonically decreasing; the list of dates must correspond to dates in the current year; the list of dates must conform to a common format.

Once we have segmented the weblog into entries, we next segment each entry into posts. We have several heuristics for finding post boundaries, such as title xpaths. If the algorithm is unable to segment the entry into posts, the entire entry is assumed to be one post.

The last step is to attempt to identify a permalink and author for each segmented post. Again, we apply an ordered set of heuristics to identify these.

The success rate of this approach is about 60% with 90% accuracy. That is, we are able to segment about 60% of weblogs into posts, and accuracy rates for the fields of the extracted posts is approximately 90%. Our main sources of error are: (1) failure to extract dates for the weblog (our extractor fails on foreign language dates); (2) parity errors that occur when our model fails to accurately represent the weblog, (e.g. when the title of the entry appears before the date of the entry); and (3) only one entry on the weblog home page.

We complement our approach to model-based segmentation using weblog feeds when available. The weblog feed contains the updated content of the weblog in standardized XML format (different flavors of RSS; Atom). A number of weblog hosting systems, such as livejournal and xanga, automatically provide a full-content feed for each hosted weblog. For such weblogs, we automatically use the feed to extract new posts with near 100% accuracy instead of crawling and segmenting the weblog. This allows us to improve our overall coverage to about 80%.

Overall, our approach to harvesting weblog posts can be summarized as follows:

1. Gather recently updated weblog URLs;
2. Automatically find feed for weblog;
3. If feed is full content, index posts from the feed;
4. Otherwise, apply model-based segmentation approach and index each extracted post.

Search over this index of weblog posts is publicly available at http://www.blogpulse.com [11].

3.1.2 Message boards

Message boards are an important communication system for tens of thousands of online communities—in fact, for many small online communities, message boards are the primary communication system.

As there is no centralized index of message boards, discovery is not trivial. We locate new boards from which to harvest by searching for keyphrases indicative of message boards on Web search engines. We then refine the search using terms indicative of a particular domain, such as automotive or gaming. In many cases, our customers also provide a list of message boards to include in harvesting.

We have implemented a system called BoardPulse for harvesting from online message boards. BoardPulse is built on two technologies—web-site wrapping and intelligent crawling [10]. Wrapping message boards is difficult for two reasons. The first issue is site complexity: while message board sites share a common structure, most boards are very complex, and many are highly customized. The second issue is one of scale. There are many thousands of different message board sites, all of which change dynamically. Message board sites cannot be efficiently crawled and indexed without detailed understanding of the structure of the site and of the mechanisms used to update the site. Acquiring and maintaining this understanding for each one of thousands of different sites is challenging.

To overcome these problems, BoardPulse exploits certain common properties that hold for most message boards. The typical message board site has a top level page listing a set of *forums*. Each forum is hyperlinked to a second level: a page (or pages) containing the set of *topics* for that forum. In turn, each topic links to a third level, the set of postings for the topic. Many large message boards also have a fourth *sub-forum* level.

Most message boards are also generated by one of a handful of message board software systems. This leads to less regularity than one would expect, however, because widely-used message board software systems are highly customizable. This customizability means that we have potentially tens of thousands of wrappers to create and maintain.

To address this issue, we use *wrapper learning* methods to reduce the cost of developing wrappers [8, 13] and the new technique of *cluster wrapping* to learn wrappers which apply to multiple message board systems. The wrapper learning system we use was also extended to take advantage of programmatic markers left in the HTML generated by message board software systems. A final property of the wrapper-learning system that we exploited was the transparency of the wrappers it produces: learned wrappers are designed to be human-readable, and can be manually modified (for instance to complete a wrapper for a cluster that could not be completely learned).

Another significant problem is how to minimize the impact of our spider on the message board servers; since many message boards are run by small communities, they often do not have the resources to allow frequent complete crawls. To address this issue, we have derived wrapper rules that extract not only data values, but also links to extract and enqueue to the spider. (The wrapping systems described in [4, 19] likewise include rules for directed crawling.) We then extended the wrapper-directed spider so that links are added to the spider queue only when two criteria hold: (1) the link matches a rule in the wrapper; and (2) a data item extracted by the wrapper has changed since the last crawl. This enables BoardPulse to perform incremental, directed crawls of message boards. BoardPulse only follows links to forums, topics, and message pages; in addition, BoardPulse only follows a link to a forum if the displayed number of posts to that forum has changed since the last crawl, and only follows a link to a topic if the displayed number of

posts to that topic has changed since the last crawl. These incremental crawling strategies all reduce the impact on the board itself.

3.2 Search Queries and Relevance

Our content system indexes hundreds of millions of internet messages. For any given project, only a small fraction of these messages are relevant. The combined purpose of search and relevance classification is to select a large portion of relevant messages from the content system while including only a small fraction of unrelated messages for analysis. The system uses a two-stage approach, combining complex boolean queries to the search engine and a machine learning relevancy classifier trained by active learning.

A well defined boolean query has a high message relevance recall at a tolerably low precision for this stage ($> 10\%$). Our system allows for six different categories of terms to be specified which are combined to construct the complex boolean query for the search. These include:

- Product terms, and Competitor terms. These words and phrases describe the main focus of the project. In general, every issue or brand that the project sets out to analyze will have some representation here in the form of one or more words or phrases. Typically, there will be a number of phrases for each issue or brand including synonyms, spelling variations, plurals, and so on.

- Source inclusion. These are message sources (boards or forums) where any message is retrieved. If a board's entire purpose is to cover the product or competitor every message from the board would be included.

- Source exclusion. These are message sources where every message is entirely excluded. For example, 'Off Topic' forums on PDA message boards might be excluded from a project about PDAs.

- Domain terms. These terms are commonly found in the domain but are not necessarily the main focus of the project. One way in which these are used is to distinguish messages that contain ambiguous product and competitor terms (e.g. distinguishing Shell the oil company from sea shells).

- Confusing terms. When the Product or Competitor terms are ambiguous, these confusing terms help exclude messages containing them from the search.

All messages retrieved by the search queries are further filtered by a machine learning text classifier. To train this classifier during the configuration process a random sample of messages matching the search query are retrieved. An analyst labels training and testing sets using an active learning process that creates both a bag-of-words classifier and precision/recall performance estimates. The active learning process incorporates a heterogeneous blend of active learning strategies that leverage domain knowledge provided by the analyst through the keyword lists above, as well as traditional active learning strategies for text classification [15, 16, 18].

Typically, the configuration of the queries and the relevance component is an iterative process. To this end, the configuration process encourages early exploration of messages and refinement of the search criteria. This helps minimizes unnecessary decision making by postponing the bulk of message labeling until a satisfactory search precision and recall are achieved. Th active learning is structured to quickly highlight poorly chosen required terms or ones that need further qualification through the use of confusion terms. Shortcuts are provided which easily enable addition or removal of query terms by selecting text in messages being inspected.

3.3 Document Analysis

Document analysis is concerned with interpreting an encoding of a document and deriving a logical structure (e.g. chapter, section, paragraph). The logical structure is generally a graph and most often a tree. Document analysis of discussion messages in web documents presents a number of interesting challenges. First, web pages (i.e. single HTML files) are different in many respects to other encodings of documents. Two of the main differences are peripheral content (e.g. navigation, adverts, branding elements) and distributed content (the document may be logically or physically broken down across many web pages). Second, the document elements (messages) that our system deals with are generally presented in a collection on a single web page. Weblogs present posts as date ordered sequences and message boards collect threads (or parts of threads) in a similar manner.[3]

Consequently, our document analysis solution really begins in the crawling stage where wrappers (either static models or inferred) are used to segment pages (see Section 3.1). This process navigates sites, removes the peripheral content and segments the web page into post granularity.

The online messages we analyze exist in a social context. Message boards and Usenet data are posted to forums or groups. They are parts of threads. Some or all of this information is encoded in the document either as explicit meta-data, or as document structure. The explicit metadata often encodes forum and group information as well as, in the case of Usenet data, link information representing the thread tree. Message boards typically have a less explicit thread structure which can be inferred from the ordering of messages (post time) and the quoted content.

We model the logical structure of a message body as a tree with the following possible node types:

- citation header
- quoted material
- signature block
- text

In addition, text blocks are segmented into paragraph blocks and, at a later stage, we segment the paragraphs and other text blocks into sentences where appropriate.

The document analysis system, designed with both efficiency and accuracy in mind, follows the explicit tree structure of the logical model - a set of analysis modules accept nodes and produce zero or more children. These analyses

[3]This issue presents a significant challenge to indexing engines such as Google that are web-page based and can not deal with sub-page indexing.

are run in a cascade, refining the output of previous analyses. An executive algorithm controls which analyses are run at which time.

A simpler system could be built with no document analysis, taking the entire text of the document as a single data type. However, the document analysis provides a number of important benefits:

- In Usenet data, quotes are represented by the convention of a distinguished symbol appearing in the left margin. As the content is preformatted, this convention inserts characters between tokens. Thus the phrase `important phrase` may be encoded as `important > phrase` confusing NLP and other sequential data analyses.
- When counting tokens, searching messages and classifying messages, it is desirable to have control over the role the tokens have in a document. For example, in certain communities, it is common for signatures to list the authors' interests or possessions (e.g. cars and video game consoles). This has a profound impact on determining what the document is about.
- It is often the case that discourse structure is encoded in the quotation structure of a document. Resolving reference requires access to this structure.

The document analysis system is built on a common framework with specialized implementations for different types of document sources, and source-dependent (and independent) analyses. Preformatted data, such as Usenet, encodes newline information at the object level whereas HTML documents encode it via the meta tags contained in the document. Specific encoding systems provide a uniform interface to certain views of the documents. For example, we can iterate over the lines in a document regardless of the underlying encoding.

Determining the quote structure requires access to the meta-tags for HTML documents. Usenet data, on the other hand, requires recognition of distinguished symbols (`>`, `|`, etc.) and an algorithm to disentangle multiple re-wrappings of lines withing the context of these symbols.

Signatures are analyzed in two ways. The simple analyses looks for signature demarcations at the bottom of text blocks: generally ASCII-art lines. A more sophisticated approach captures signatures that do not follow this type of pattern. We take a set of documents and look for repeated content across messages at the end of messages or quoted text blocks (cf [6]). In this way, we develop a database of unique signature patterns and are able to tag a signature without explicit boundary markers if it occurs multiple times in the data.

3.4 Topic Classification

In a marketing intelligence application of data mining, there are typically topics of discussion in the data that warrant explicit tracking and identification. The most prevalent type of topics are brand-related, i.e. one topic for each product or brand being tracked, such as the *Dell Axim*. To facilitate this taxonomic requirement, analysts compose well-written hand-built rules to identify these types of topics. These rules are based on words and phrases, and allow for stemming, synonymy, windowing, and context-sensitivity based on document analysis.

From one point of view, these brands are entities occurring in the text, and it might be considered that entity extraction would be the most appropriate technology to apply. However, to facilitate tracking and identification, extracted entities must be normalized to a set of topics. For example, *Axim*, *Dell Axim*, and *the Dell PDA* should all fall into the Dell Axim topic. An approach following that of [7] could be established to automatically normalize entities. However, since our customers typically know exactly which brands they want to monitor, pre-building the rules in this case is both more accurate and the performance is more predictable and can be easily measured.

In addition to brand-like topics defined through rules, it's often the case that other topics are more accurately recognized from a complex language expression that is not easily captured by a rule. For example, topics such as *Customer Service* are not so simply captured by sets of words, phrases and rules. Thus, we often approach topic classification with machine learning techniques. The provided classifier is trained with machine learning techniques from a collection of documents that have been hand-labeled with the binary relation of topicality. The hand-labeling by the analysts is performed using an active learning framework (similar to Section 3.2). The underlying classifier is a variant of the Winnow classifier [17], an online learning algorithm that finds a linear separator between the class of documents that are topical and the class of documents that are irrelevant. Documents are modeled with the standard bag-of-words representation that discards the ordering of words and notices only whether or not a word occurs in a document. Empirically, we have found Winnow to be a very effective document classification algorithm, rivaling the performance of Support Vector Machines [14] and k-Nearest Neighbor [26], two other state-of-the-art text classification algorithms. This machine learning classification and application is described more fully in [12].

3.5 Polarity

The detection of sentiment, or polarity, in text is an area of research gaining considerable momentum ([24]). Broadly speaking there are three main approaches described in the current literature. Firstly, methods which build on document classification methods [23]. Here the features used by the system are features of the text (unigram, bigrams, etc.) and supervised machine learning algorithms are trained on some collection of labeled data. Secondly, there are those methods which use linguistic analysis of some type [21]. These approaches often employ a lexicon of important terms and shallow parsing methods. Thirdly, there are those approaches which aim to use aggregate social cues from the context within which documents are published [2]. The approach described here is of the second type.

Polarity analysis (as we will refer to this task) is concerned with determining whether or not a piece of text describes some topic favorably or unfavorably. For example `the game was incredible` is a favorable description, `the car steers shakily` is an unfavorable one. In many contexts there are two types of polarity. Firstly, expressions which refer to emotional state (e.g. `I hated that film`). Secondly, expressions which refer to a state of affairs that is generally accepted as favorable or unfavorable (e.g. `The tire blew out on me`). This distinction is made as the majority of work on sentiment refers to the class of emotive expressions,

and not those expressions that may be termed objective, but which have a generally accepted negative orientation, such as `The computer crashed`.

There are many syntactic, semantic and discourse level constraints which effect the interpretation of polarity, including:

- Negation: `it is not good`.
- Future state and modality: `I might like it`.
- Transfer of polarity: compare `I didn't say it was good` and `I didn't hear it was good`.

The polarity module consists of the following elements:

- A lexicon
- A POS (part-of-speech) tagger
- A shallow parser
- Semantic rules

In developing the POS tagger, we encountered two significant issues. Firstly, the standard training sets used in the literature for training do not cover the online text or document genres that we are working with. Most importantly, for terms with multiple possible tags, the distribution of term/tag pairs is often quite different. `like` appears in the WSJ most often as a preposition. However, in our data `like` appears mostly as a verb. Certain senses of this verb, of course, carry polar meaning. To deal with this problem we had to create our own auxiliary data to train the tagger.

The second problem, and again, one which distinguishes our tagger from the standard paradigm, is our internal model of the object data. The standard paradigm is to accept a string, partition this string into tokens (which we might call words) and tag the words. However, in the genre of text that we are dealing with, this model is not suitable. For example, in the segment `ill buy a new one` there is no single tag that can be applied to the token `ill`. This token cannot be split arbitrarily (into `i` and `ll`) due to the ambiguity with that token as a single word (the adjective indicating poor health). Consequently, the model of text that we work with considers the text layer as a signal generated by a sequence of words. Our goal is to tag this underlying sequence of words, not a partitioning of the text generated from those words by a tokenizer. We use hand crafted rules to recover the words and are currently formalizing this approach.

The shallow parser we use is a cascade of transducers. Effectively, each cascade may build internal structure. The structure built is similar to a phrase marker, though is not constrained to capture grammatical structure per se ([1]).

Once this approximate grammatical structure is derived, the semantics of the expression is computed in a bottom up compositional manner resulting in a polarity feature for the span of text. The features and rule application for polarity extraction is described in full detail in [22].

3.6 Fact Extraction

Having each message tagged according to the topics and polarity identified within the message allows for some types of analysis. However, a message-level tagging does not allow any conclusions to be drawn about the intersections of topics and sentiment. For example, a message that is `positive` and contains the topics of `Dell Axim` and `Display` does not necessarily say anything positive about Dell Axim's display. To facilitate this, a further analysis of fact extraction is layered on top of the sentiment and topic analysis to understand at a finer level the expressions made within a message.

In previous work [12] we showed that in the domain of online message discussion, intersecting sentiment with topic classifiers at the sentence level provides precision around 65%. We extend this same approach to intersections of sentiment with multiple topics at a time. However, relying on message intersection provides fairly low recall. To increase this recall, we use simple resolution techniques to associate brand-like topics (e.g. `Dell Axim`) with topics describing features of brands (e.g. `Customer Service` or `Peripherals`). For example, a brand can be referenced in the Subject line of a blog, and feature-like topics mentioned in the body of the blog resolve back to the brand topics in the subject line when other brands are not mentioned in the body. In this way, we identify facts that can be thought of as triples of brands, their (optional) features, and the (optional) polarity of the authorial expression. Each fact is backed by a segment of text (a sentence or a paragraph) that can be used for finer-grained analysis during interactive use of the system.

Fact extraction is the culmination of the content system and the production system configured for a specific domain. Retrieved relevant messages are tagged with topics and then analyzed for sentiment and combination of brand topics with other topics. At this point, these extracted facts could be exported to a traditional data mining system. However, since each fact is backed by a segment of text, advanced text data mining algorithms are more appropriate for analysis. The case study in Section 2 gave some examples of specific text analyses that led to marketing intelligence. The next section describes some of these technologies in more detail.

4. INTERACTIVE DATA ANALYSIS

We have designed our analysis tool around two simple concepts: data selection and data viewing. On top of this, we provide a powerful pervasive capability: any view offers standard mechanisms to further refine the data selection - drill-down. For example, when viewing a message we can highlight a word and click through. This will segment the data to include only those messages that contain that word. This principle provides a key interface strategy in the battle against complexity: predictable and intuitive mechanisms available through consistent interactions at any time.

The data selection mechanism (slicing) essentially builds a tree of filters. These filters (e.g. relevance, topic, phrase, etc.) are applied in sequence resulting in a current data set of facts and messages. Forward and backwards buttons supply browsing capabilities similar to a web browser and a history panel provides the complete data selection history.

The currently selected set of facts and messages can be applied to a variety of data exploration and analyses. Some of these are straightforward, such as keyword-in-context, and full display of all messages or facts. Others are reminiscent of traditional data analysis, such time series analysis. Others, described in this section, leverage the unique text characteristics of the data.

4.1 Phrase Finding

Suppose we have identified that a certain product has a lot of negative comments associated with it, and would like

to quickly know what issues people are mentioning in those messages. When the volume of the target set of messages is large, browsing messages is not an efficient way to understand the contents of the messages.

Phrase finding, which enables the user to identify key concepts by browsing a list of automatically extracted phrases, is a useful tool for such situations. There are three types of data-oriented phrase finding capabilities in the system:

1. Given a set of messages, find keyphrases which are commonly mentioned in the messages.
2. Given two sets of messages, find the set of keyphrases that best discriminate the two sets.
3. Given a phrase and surrounding context from a set of messages, find collocations (words or phrases which frequently appear together with the specified phrase).

One of the challenges in extracting an *informative* set of phrases is that a frequent word or phrase is not necessarily a good keyphrase. If we simply extract frequent words or phrases, you end up with function words or idiomatic phrases. To capture informativeness, we make use of the relationship between a *foreground* and a *background* corpus.

The target document set from which keyphrases are extracted is called the foreground corpus. The document set to which this target set is compared is called the background corpus. Examples of foreground and background corpora include: a web site of a company and web data in general; a newsgroup and the whole Usenet archive; and research papers of a certain conference and research papers in general.

For our example of extracting keyphrases in Table 1, the background corpus is the set of messages about the Axim and the foreground corpus is the paragraphs in these messages having negative polarity about the Axim. Our system enables us to quickly set both foreground and background corpora simply by double-clicking a table row, selecting a time range, or selecting a cluster in a social network graph.

The collocation extraction algorithm also uses the foreground and background corpus using the local contexts in which the target phrase appears as the foreground corpus. In the workbench, the collocation extraction mechanism is integrated into the Keyword In Context (KWIC) analysis. This enables the user to select the width of the target context interactively and try various collocation metrics.

A phrase finder is typically a pipeline of phrase finder components. A phrase finder component takes a foreground corpus and optionally a background corpus and/or a list of seed phrases, and returns a list of phrases together with an associated score for each phrase. A seeded phrase finder component may be implemented to act as filters and rescorers, as well as to provide methods to extend phrases in phrase list or expand the phrase list in some way. The following is the typical pipeline we use to extract key noun phrases:

1. A KeyBigramFinder, which takes foreground and background corpora and returns informative bigrams. The key is to combine a measure of *informativeness* and a measure of *phraseness* for a bigram into a single unified score to produce a ranked list of key-bigrams. One of the methods we use to extract informative bigrams is described in [25].
2. An AprioriPhraseExpander, which takes the top N phrases from a KeyBigramFinder and expands it into longer phrases that occurs more than M times. It uses a priority queue of phrases sorted by frequency and heuristics for generating expansion candidates, similar to the APRIORI algorithm [3]. Sentence and block boundaries and the linguistic class of a token is checked to see if a candidate phrase can be expanded or not.
3. A ConstituentFilter is used when we want to extract only noun phrases. It checks occurrences of a phrase in the data to find contextual evidence that the phrase is a noun phrase.

The resulting phrase list is sorted by either document frequency or by an informativeness score and presented as the results of the analysis.

Efficiency for phrase finding is very important, since results are computed in real-time during interactive analysis. The backing data structure to facilitate efficient phrase finding we call a corpus. A corpus is a collection of tokenized messages, which is derived from the result of the document analysis step described in Section 3.3 by applying paragraph and sentence segmenter, tokenizer, then applying part-of-speech tagger over the resulting token sequence. After upper/lower case is normalized, the token is looked up in a symbol dictionary and a tokenized message is represented as a sequence of integers. Source information of a token such as document analysis result (e.g. within a quoted text or signature block), original case information, sentence/paragraph boundaries, and part-of-speech tags, are stored as a set of *annotations* into the corpus. This enables one to extract phrases only from unquoted text and to use part-of-speech information for extracting phrases, for example.

An inverted index is also created for each corpus, which returns document IDs and offset positions a word or phrase occurs. This allows phrase finders to inspect the corpus-wide nature of phrase candidates quickly.

4.2 Metrics

To facilitate top-down exploration of data, a number of metrics have been created that provide a high-level summary of the relevant online discussion across a number of dimensions. The key base metrics we provide are:

- Buzz Count. A simple count of the number of messages, alternately expressed as a percentage.
- Polarity. A 1-10 score representing the overall sentiment expressed about a topic or intersection of topics. The score is based on the posterior estimate of the ratio of the frequency of positive to negative comments. It is described more fully in [22].
- Author Dispersion. A measure of how spread out the discussion of a particular topic is. High values indicate that many people are talking about a particular topic, where low values indicate that discussion is centered around a small group of people. This measure is more indicative than just counting of unique authors for a topic, as error in the topic classifications dilutes the understanding of the spread of discussion.
- Board Dispersion. Similar to author dispersion, this measures how many different places are seeing discussion about a particular topic. Topics that have a board

dispersion that grows rapidly over time indicates a viral issue. If such a viral issue is negative, prompt attention is often recommended.

These metrics serve two purposes. First, they give a starting point for top-down exploration. Second, they provide dashboard-style summary statistics that can be disseminated within an organization, tracked over time, and monitored for improvement or directionality.

5. CONCLUSION

Online discussion, in the form of blogs and boards, represents a valuable opportunity for many types of analyses. This paper has described an end-to-end system that gathers specific types of online content and delivers analytics based on classification, NLP, phrase finding and other mining technologies in a marketing intelligence application.

The analysis system allows a user to rapidly characterize the data and drill down to discover and validate specific issues. The system delivers both qualitative and quantitative accounts of features derived from online messages.

6. REFERENCES

[1] S. Abney. Partial parsing via finite-state cascades. In *Workshop on Robust Parsing, 8th European Summer School in Logic, Language and Information*, 1996.

[2] R. Agrawal, S. Rajagopalan, R. Srikant, and Y. Xu. Mining newsgroups using networks arising from social behavior. In *Proceedings of the Twelfth International World Wide Web Conference (WWW2003)*, 2003.

[3] R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In J. B. Bocca, M. Jarke, and C. Zaniolo, editors, *Proc. 20th Int. Conf. Very Large Data Bases, VLDB*, pages 487–499. Morgan Kaufmann, 12–15 1994.

[4] R. Baumgartner, S. Flesca, and G. Gottlob. Declarative information extraction, Web crawling, and recursive wrapping with Lixto. *Lecture Notes in Computer Science*, 2173, 2001.

[5] K. D. Bollacker, S. Lawrence, and C. L. Giles. CiteSeer: An autonomous web agent for automatic retrieval and identification of interesting publications. In *Agents '98*, pages 116–123, 1998.

[6] H. Chen, J. Hu, and R. W. Sproat. Integrating geometric and linguistic analysis for e-mail signature block parsing. *ACM Transactions on Information Systems*, 17(4):343–366, 1999.

[7] W. W. Cohen. Data integration using similarity joins and a word-based information representation language. *ACM Transactions on Information Systems*, 18(3):288—321, 2000.

[8] W. W. Cohen, L. S. Jensen, and M. Hurst. A flexible learning system for wrapping tables and lists in HTML documents. In *Proceedings of The Eleventh International World Wide Web Conference (WWW-2002)*, Honolulu, Hawaii, 2002.

[9] M. Craven, D. DiPasquo, D. Freitag, A. McCallum, T. Mitchell, K. Nigam, and S. Slattery. Learning to construct knowledge bases from the World Wide Web. *Artificial Intelligence*, 118(1–2):69–113, 2000.

[10] N. Glance and W. Cohen. BoardViewer: Meta-search and community mapping over message boards. Intelliseek Technical Report, 2003.

[11] N. Glance, M. Hurst, and T. Tomokiyo. BlogPulse: Automated trend discovery for weblogs. In *WWW 2004 Workshop on the Weblogging Ecosystem: Aggregation, Analysis and Dynamics*, 2004.

[12] M. Hurst and K. Nigam. Retrieving topical sentiments from online document collections. In *Document Recognition and Retrieval XI*, pages 27–34, 2004.

[13] L. S. Jensen and W. Cohen. Grouping extracted fields. In *Proceedings of the IJCAI-2001 Workshop on Adaptive Text Extraction and Mining*, 2001.

[14] T. Joachims. Text categorization with support vector machines: Learning with many relevant features. In *Machine Learning: ECML-98, Tenth European Conference on Machine Learning*, 1998.

[15] D. D. Lewis and J. Catlett. Heterogeneous uncertainty sampling for supervised learning. In *Machine Learning: Proceedings of the Eleventh International Conference*, 1994.

[16] D. D. Lewis and W. A. Gale. A sequential algorithm for training text classifiers. In *SIGIR '94*, pages 3–12, 1994.

[17] N. Littlestone. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Machine Learning*, 2:285–318, 1988.

[18] A. McCallum and K. Nigam. Employing EM in pool-based active learning for text classification. In *Machine Learning: Proceedings of the Fifteenth International Conference*, pages 350–358, 1998.

[19] J. Myllymaki. Effective web data extraction with standard XML technologies. In *Proc. WWWW10*, pages 689–696, May 2001.

[20] T. Nasukawa, M. Morohashi, and T. Nagano. Customer claim mining: Discovering knowledge in vast amounts of textual data. Technical report, IBM Research, Japan, 1999.

[21] T. Nasukawa and J. Yi. Sentiment analysis: Capturing favorability using natural language processing. In *Proceedings of K-CAP '03*, 2003.

[22] K. Nigam and M. Hurst. Towards a robust metric of opinion. In *AAAI Spring Symposium on Exploring Attitude and Affect in Text*, 2004.

[23] B. Pang, L. Lee, and S. Vaithyanathan. Thumbs up? sentiment classification using machine learning techniques. In *Proceedings of EMNLP 2002*, 2002.

[24] J. G. Shanahan, Y. Qu, and J. Weibe, editors. *Computing Attitude and Affect in Text*. Springer, Dordrecht, Netherlands, 2005.

[25] T. Tomokiyo and M. Hurst. A language model approach to keyphrase extraction. In *Proceedings of the ACL Workshop on Multiword Expressions*, 2003.

[26] Y. Yang. An evaluation of statistical approaches to text categorization. *Information Retrieval*, 1(1/2):67–88, 1999.