In *Proceedings of the International Conference on Discovery Science.*2001.

Clipping and Analyzing News Using Machine Learning Techniques

Hans Gründel, Tino Naphtali, Christian Wiech,
Jan-Marian Gluba, Maiken Rohdenburg, and Tobias Scheffer

SemanticEdge, Kaiserin-Augusta-Allee 10-11, 10553 Berlin, Germany
{hansg, tinon, christianw, jang, scheffer}@semanticedge.com

Abstract. Generating press clippings for companies manually requires a considerable amount of resources. We describe a system that monitors online newspapers and discussion boards automatically. The system extracts, classifies and analyzes messages and generates press clippings automatically, taking the specific needs of client companies into account. Key components of the system are a spider, an information extraction engine, a text classifier based on the Support Vector Machine that categorizes messages by subject, and a second classifier that analyzes which emotional state the author of a newsgroup posting was likely to be in. By analyzing large amount of messages, the system can summarize the main issues that are being reported on for given business sectors, and can summarize the emotional attitude of customers and shareholders towards companies.

1 Introduction

Monitoring news paper or journal articles, or postings to discussion boards is an extremely laborious task when carried out manually. Press clipping agencies employ thousands of personnel in order to satisfy their clients' demand for timely and reliable delivery of publications that relate to their own company, to their competitors, or to the relevant markets. The internet presence of most publications offers the possibility of automating this filtering and analyzing process. One challenge that arises is to analyze the content of a news story well enough to judge its relevance for a given client. A second difficulty is to provide appropriate overview and analyzing functionality that allows a user to keep track of the key content of a potentially huge amount of relevant publications.

Software systems that spider the web in search of relevant information, and extract and process found information are usually referred to as *information agents* [16, 2]. They are being used, for instance, to find interesting web sites or links [19, 12], or to filter news group postings (*e.g.,* [26]). One attribute of information agents is how they determine the relevance of a document to a user.

Content-based recommendation systems (*e.g.,* [1]) judge the interestingness of a document to the user based on the content of other documents that the user has found interesting. By contrast, collaborative filtering approaches (*e.g.,* [13]), draw conclusions based on which documents other users with similar preferences have found interesting. In many applications, it is not reasonable to ask the user to elaborate his or her preferences explicitly. Therefore, information agents often try to *learn* a function that expresses user interest from user feedback; *e.g.,* [26, 18]. By contrast, a user who approaches a press clipping agency usually has specific, elaborated information needs.

The problem of identifying predetermined relevant information in text or hypertext documents from some specific domain is usually referred to as *information extraction* (IE) (*e.g.,* [4, 3]). In the news clipping context, several instances of the information extraction problem occur. Firstly, press articles have to be extracted from HTML pages where they are usually embedded between link collections, adverts, and other surrounding text. Secondly, named entities such as companies or products have to be identified and extracted and, thirdly, meta-information such as publication dates or publishers need to be found.

While first IE algorithms were hand-crafted sets of rules (*e.g.,* [7]), algorithms that learn extraction rules from hand-labeled documents (*e.g.,* [8, 14, 6]) have now become standard. Unfortunately, rule-based approaches sometimes fail to provide the necessary robustness against the inherent variability of document structure, which has led to the recent interest in the use of Hidden Markov Models (HMMs) [25, 17, 21, 23] for this purpose.

In order to identify whether the content of a document matches one of the categories the user is interested in and to summarize the subjects of large amounts of relevant documents, classifiers that are learned from hand-labeled documents (*e.g.,* [24, 11]) provide a means of categorizing a document's content that reaches far beyond key word search. Furthermore, it can be interesting to determine the emotional state [9] of authors of postings about a company or product.

In this paper, we discuss a press clipping information agent that downloads news stories from selected news sources, classifies the messages by subject and business sector, and recognizes company names. It then generates customized clippings that match the requirement of clients. We describe the general architecture in Section 2, and discuss the machine learning algorithms involved in Section 3. Section 4 concludes.

2 Publication Monitoring System

Figure 1 sketches the general architecture of the system. A user can configure the information service by providing a set of preferences. These include the names of all companies that he or she would like to monitor, as well as all business areas (*e.g.,* biotechnology, computer hardware) of interest The spider cyclically downloads a set of newspapers, journals, and discussion boards. The set of news sources is fixed in advance and not depending on the users' choices. All downloaded messages are recorded in a news database after the extraction engine has stripped the HTML code in which the message is embedded (header and footer parts as well as HTML tags, pictures, and advertisements).

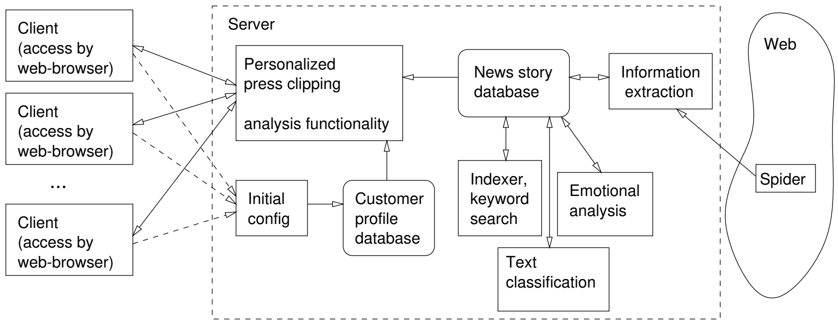

Fig. 1. Overview of the SemanticEdge Publication Monitoring System

The spider developed by SemanticEdge is configured by providing a set of patterns which all URLs that are to be downloaded have to match. Typically, online issues of newspapers have a fairly fixed site structure and only vary the dates and story numbers in the URLs daily. Depending on the difficulty of the site structure, configuring the spider such that all current news stories but no advertisements, archives, or documents that do not directly belong to the newspaper are downloaded, requires between one and four hours.

Text classifier, named entity recognizer, and emotional analyzer operate on this database. The text classifier categorizes all news stories and newsgroup postings whereas the emotional analyzer is only used for newsgroup postings; it classifies the emotional state that a message was likely to be written in. For each client company, a customized press clipping is generated, including summarization and visualization functionality.

The press clipping consists of a set of dynamically generated web pages that a user can view in a browser after providing a password. The system visualizes the number of publications by source, by subject, and by referred company. For each entry, an emotional score between zero (very negative) and one (very positive) is visualized as a red or green bar, indicating the attitude of the article (Fig. 2), or the set of summarized articles. Figure 2 shows the list of all articles relevant to a client, Figure 3 shows the summary mode in which the system summarizes all articles either from one news source, or about one company, or related to one business sector per line. The average positive or negative attitude of the articles summarized in one line is visualized by a red or green bar.

Several diagrams visualize the frequency of referrals to business sectors or individual companies and the average expressed attitude (Figure 4).

3 Intelligent Document Analysis

Document analysis consists of information extraction (including recognition of named entities), subject classification, and emotional state analysis.

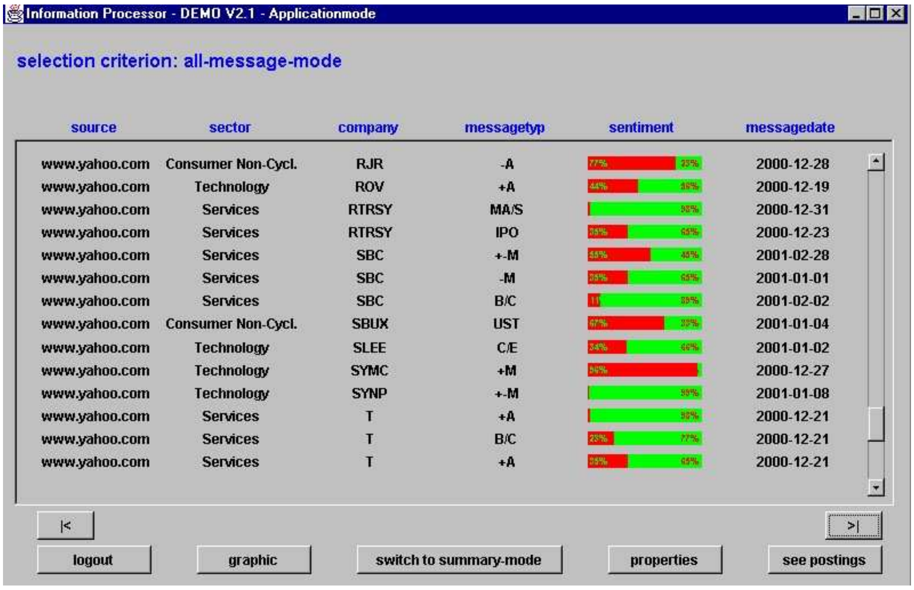

Fig. 2. Press clipping for client company: message overview

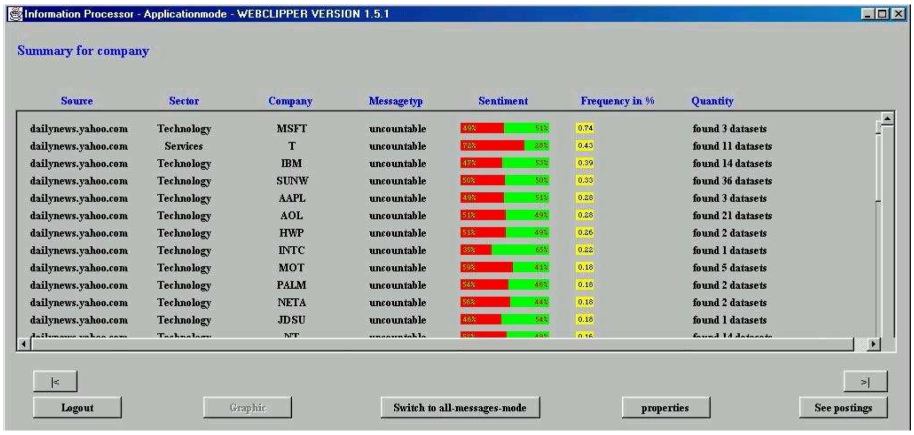

Fig. 3. Press clipping for client company: company summary

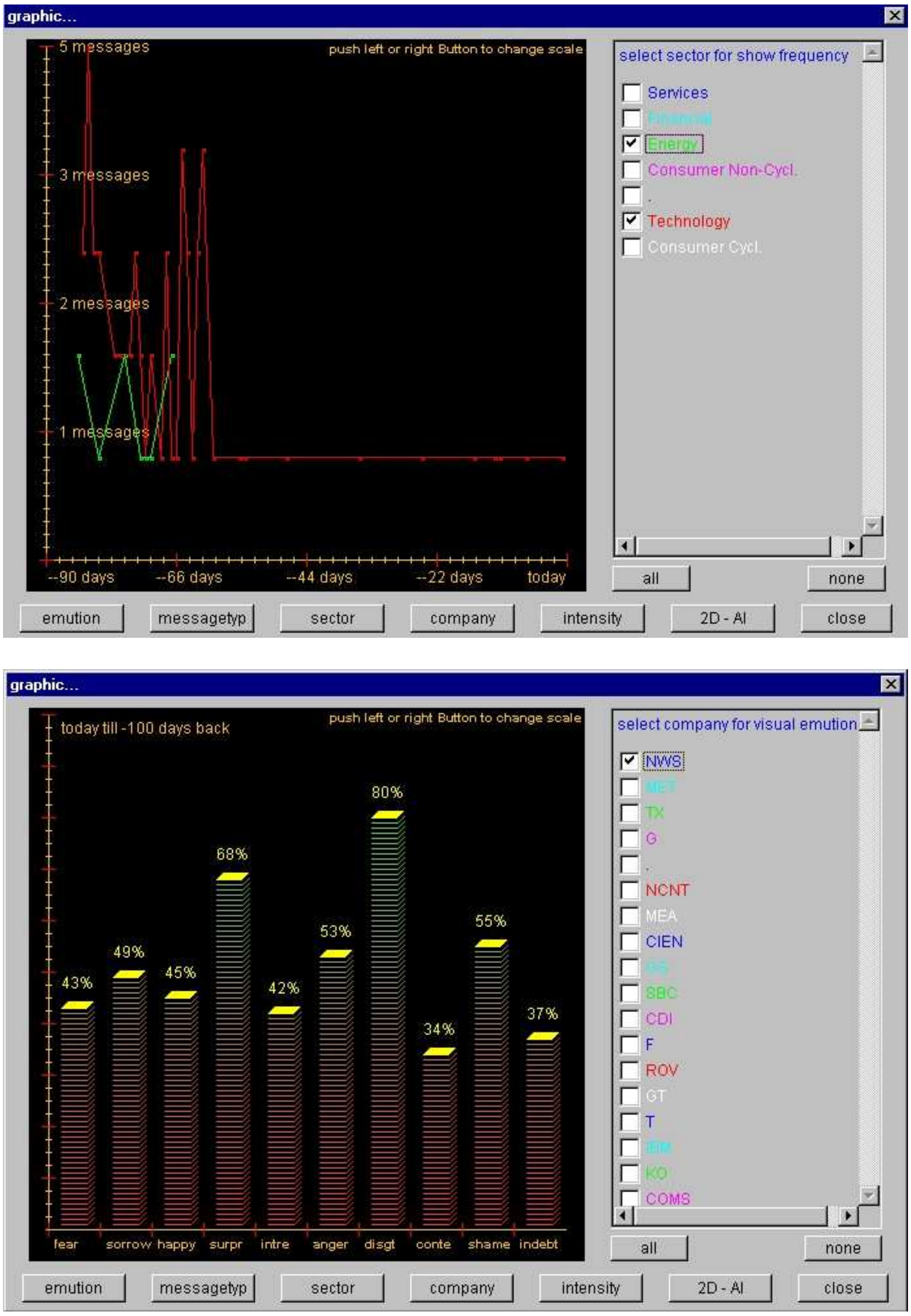

Fig. 4. Top: frequency of messages related to business sectors. Bottom: expressed emotional attitude toward companies

3.1 Information Extraction

Two main paradigms of information extraction agents which can be trained from hand-labeled documents exist; algorithms that learn extraction rules (*e.g.,* [8, 14, 6]) and statistical approaches such as Markov models [25, 17], partially hidden Markov models [21, 23] and conditional random fields [15].

Rule base information extraction algorithms appear to be particularly suited to extract text from pages with a very strict structure and little variability between documents. In order to learn how to extract the text body from the HTML page of a Yahoo! message board, the proprietary rule learner that we use needs only one example in order to identify where, in the document structure, the information to be extracted is located. We can then extract text bodies from other messages with equal HTML structure with an accuracy of 100%.

Many other information extraction tasks, such as recognizing company names, or stock recommendations, rule based learners do not provide enough robustness to deal with the high variability of natural language. Hidden Markov models (HMMs) (see, [20] for an introduction) are a very robust statistical method for analysis of temporal data. An HMM consists of finitely many states $\{S_1, \ldots, S_N\}$ with probabilities $\pi_i = P(q_1 = S_i)$, the probability of starting in state S_i, and $a_{ij} = P(q_{t+1} = S_j | q_t = S_i)$, the probability of a transition from state S_i to S_j. Each state is characterized by a probability distribution $b_i(O_t) = P(O_t | q_t = S_i)$ over observations. In the information extraction context, an observation is typically a token. The information items to be extracted correspond to the n target states of the HMM. Background tokens without label are emitted in all HMM states which are not one of the target states.

HMM parameters can be learned from data using the Baum-Welch algorithm. When the HMM parameters are given, then the model can be used to extract information from a new document. Firstly, the document has to be transformed into a sequence of tokens; for each token, several attributes are determined, including the word stem, part of speech, the HTML context, attributes that indicates whether the word contains letter, digits, starts with a capital letter and other attributes. Thus, the document is transformed into a sequence of attribute vectors.

Secondly, the forward-backward algorithm [20] is used to determine, for each token, the most likely state of the HMM that it was emitted in. If, for a given token, the most likely state is one of the background states, then this token can be ignored. If the most likely state is one of the target states and thus corresponds to one of the items to be extracted, then the token is extracted and copied into the corresponding database field.

In order to adapt the HMM parameter, a user first has to label information to be extracted manually in the example document. Such partially labeled documents form the input to the learning algorithm which then generates the HMM parameters. We use a variant of the Baum-Welch algorithm [23] to find the model parameters which are most likely to produce the given documents and are consistent with the labels added by the user.

Figure 5 shows the GUI of the SemanticEdge information extraction environment. HMM based and rule based learners are plugged into the system.

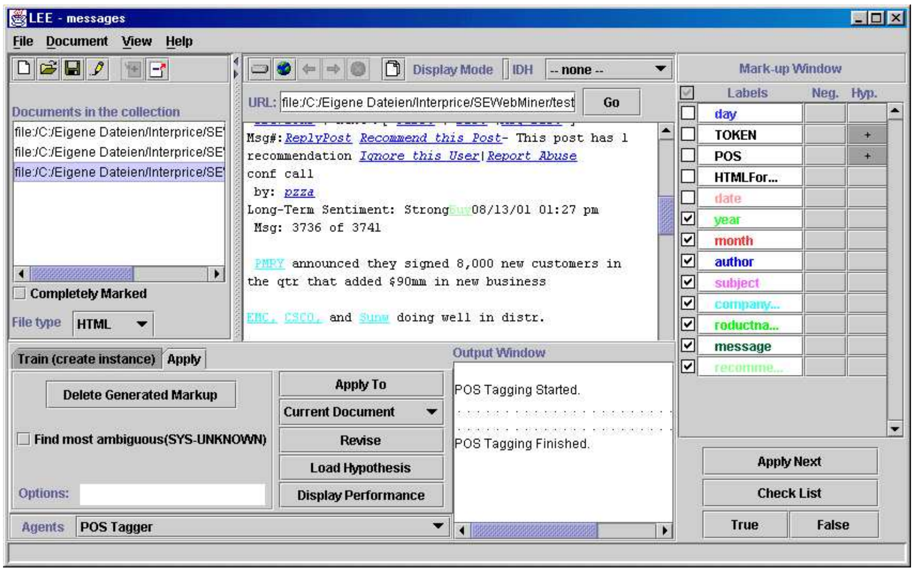

Fig. 5. GUI of the information extraction engine

For specialized information extraction tasks such as finding company names in news stories, specially tailored information extraction agents outperform more general approaches such as HMMs. For instance, most companies that are being reported about are listed at some stock exchange. To recognize these companies, we only need to maintain a dynamically growing database.

3.2 Subject Classification

For the subject classification step, we have defined a set of message subject categories (*e.g.*, IPO announcement, ad hoc message) and a set of business sector and markets categories. The resulting classifiers assign each message a set of relevant subjects and sectors.

The classifier proceeds in several steps. First, a text is tokenized and the resulting tokens are mapped to their word stems. We then count, for each word stem and each example text, how often that word occurs in the text. We thus transform each text into a feature vector, treating a text as a bag of words. Finally, we weight each feature by the inverse frequency of the corresponding word which has generally been observed to increase the accuracy of the resulting classifiers (*e.g.,* [10, 22]). This procedure maps each text to a point in a high-dimensional space.

The Support Vector Machine (SVM) [11] is then used to efficiently find a hyper-plane which separates positive from negative examples, such that the margin between any example and the plane is maximized. For each category we thus obtain a classifier which can then take a new text and map it to a negative or positive values, measuring the document's relevance for the category.

During the application phase, the support vector machine returns, for each category, a value of its decision function, that can range from large negative to large positive values. It is necessary to define a threshold value from which on a document is considered to belong to the corresponding category. There are several criteria by which this threshold can be set; perhaps the most popular is the precision recall breakeven point. The precision quantifies the probability of a document really belonging to a class given that it is predicted to lie in that class. Recall, on the other hand quantifies how likely it is that a document really belonging to a category is in fact predicted to be a member of that class by the classifier. By lowering the threshold value of the decision function we can increase recall and decrease precision, and vice versa. The point at which precision equals is often used as a normalized measure of the performance of classification and IR methods. Varying the threshold leads to precision and recall curves. Figure 6 shows the GUI of our text SVM-based categorization tool.

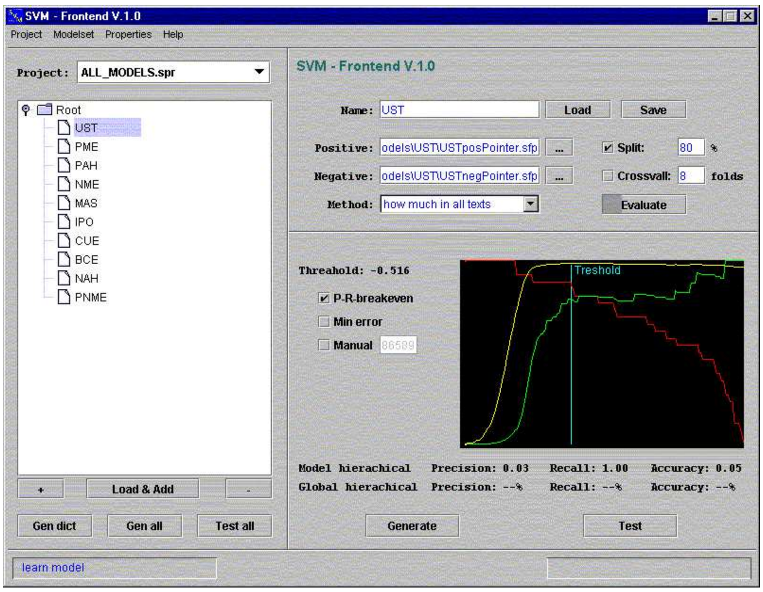

Fig. 6. GUI of the text classification engine

It is also possible to define the accuracy (the probability of the classifier making a correct prediction for a new document) as a performance measure. Unfortunately, many categories (such as IPO announcement) are so infrequent, that a classifier which in fact *never* predicts that a document does belong to this class can achieve an accuracy of as much as 99.9%. This renders the use of accuracy as a performance metric less suited than precision/recall curves.

For each category, we manually selected about 3000 examples, between 60 and 700 of these examples were positives. Figure 7 shows precision, recall, and accuracy of some randomly selected classes over the threshold value. The curves are based on hold-out testing on 20% of the data. Note that, for many of these classes such as xxx, the prior ratio of positive examples is extremely small (such as 1.4%). Specialized categories, such as "initial public offering announcement" can be recognized almost without error; "fuzzy" concepts like "positive marked news" impose greater uncertainties.

3.3 Emotional Classification

In psychology, a space of universal, culturally independent base emotional states have been identified according to the differential emotional theory (*e.g.,* [5, 9]); ten clusters within this emotional space are generally considered base emotions. These are interest, happiness, surprise, sorrow, anger, disgust, contempt, shame, fear, guilt (Figure 4).

While it is typically impossible to analyze the emotional state of the author of a sober newspaper article, authors of newsgroup often do not conceal their emotions. Given a posting, we use an SVM to determine, for each of the ten emotional states, a score that rates the likelihood of that emotion for the author. We average the scores over all postings related to a company, or to a product, and visualize the result as in Figure 4. We can project emotional scores onto a "positive-negative" ray and visualize the resulting score as a red or green bar as in Figure 2.

We manually classified postings to discussion boards into positive, negative, and neutral for each of the ten base emotional states. Emotional classification of messages turned out to be a fairly noisy process; the judgment on the emotional content of postings usually varies between individuals. Unfortunately, we found no positive examples for disgust but between 2 and 21 positive and between 16 and 92 negative examples for the other states.

Figure 8 shows precision and recall curves for those emotions for which we found most positive examples, based on 10-fold cross validation. As we expected, recognizing emotions seems to be a very difficult task; in particular, from the small samples available. Still the recognizer performs significantly better than random guessing. Emotional classifiers with a rather high threshold can often achieve reasonable precision values. Also, in many cases in which human and classifier disagree, it is not easy to tell whether human or classifier are wrong.

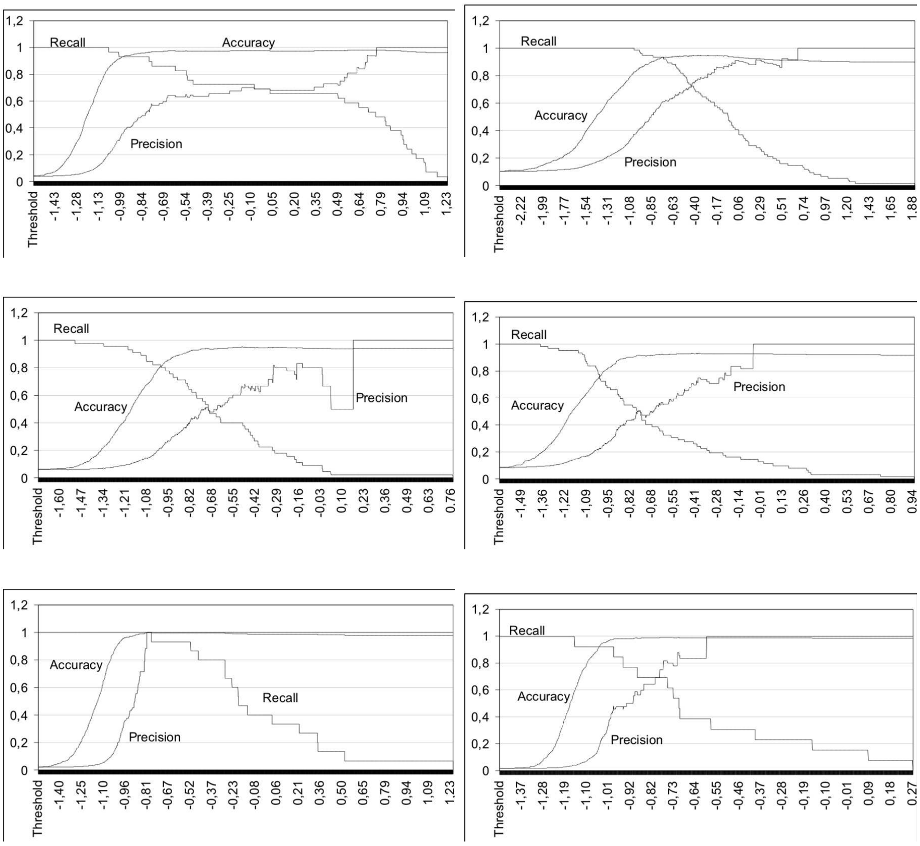

Fig. 7. Precision, recall, and accuracy for subject classification. First row: “US treasury”, “mergers and acquisition”; second row: “positive” / “negative market and economy news”; third row: “initial public offering announcement”, “currency and exchange rates”.

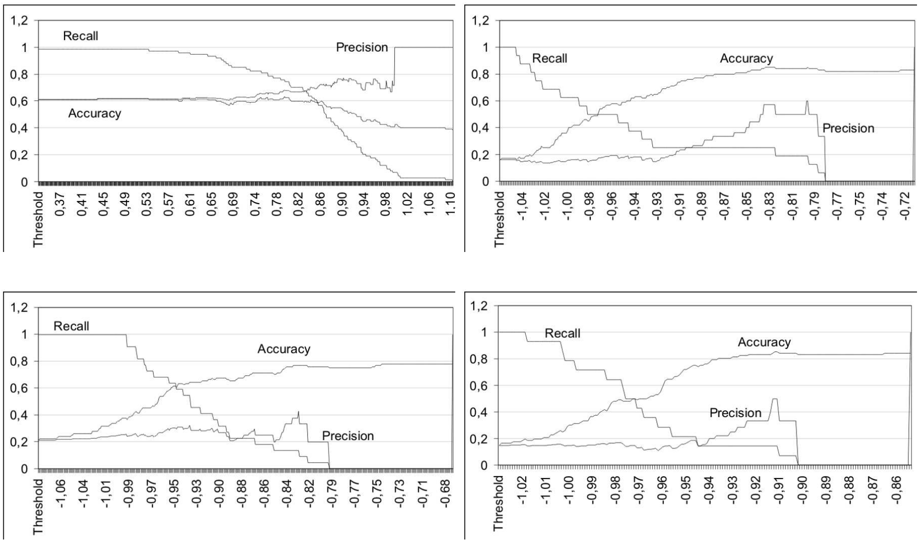

Fig. 8. Precision, recall, and accuracy for emotional classification. First row: positive versus negative, anger; second row: contempt, fear.

4 Conclusion

We describe a system that monitors online news sources and discussion boards, downloads the content regularly, extracts the document bodies, analyzes messages by content and emotional state, and generates customer-specific press clippings. A user of the system can specify his or her information needs by entering a list of company names (*e.g.,* the name of the own company and relevant competitors) and selecting from a set of message types and business sectors. Information extraction tasks are addressed by rule induction and hidden Markov modes; the Support Vector Machine is used to learn classifiers from hand-labeled data. The customer-specific news stories are listed individually, as well as summarized by several criteria. Diagrams visualize how frequently business sectors or companies are cited over time.

The resulting press clippings are generated in near real-time and fully automatically. This tool enables companies to keep track of how they are being perceived in news groups and in the press. It is also inexpensive compared to press clipping agencies. On the down side, the system is certain to miss all news stories that only appear in printed issues. Also, the classifier has a certain inaccuracy which imposes the risk of missing relevant articles. Of course, this risk is also present with press clipping agencies. Nearly all studied subject categories can be recognized very reliably using support vector classifiers.

References

1. A. Amrodt and E. Plaza. Cased-based reasoning: foundations, issues, methodological variations, and system approaches. *AICOM*, 7(1):39–59, 1994.
2. N. Belkin and W. Croft. Information filtering and information retrieval: Two sides of the same coin? *Communications of the ACM*, 35(12):29–38, 1992.
3. Mark Craven, Dan DiPasquo, Dayne Freitag, Andrew K. McCallum, Tom M. Mitchell, Kamal Nigam, and Seán Slattery. Learning to construct knowledge bases from the World Wide Web. *Artificial Intelligence*, 118(1-2):69–113, 2000.
4. L. Eikvil. Information extraction from the world wide web: a survey. Technical Report 945, Norwegian Computing Center, 1999.
5. P. Ekman, W. Friesen, and P. Ellsworth. *Emotion in the human face: Guidelines for research and integration of findings*. Pergamon Press, 1972.
6. G. Grieser, K. Jantke, S. Lange, and B. Thomas. A unifying approach to html wrapper representation and learning. In *Proceedings of the Third International Conference on Discovery Science*, 2000.
7. Ralph Grishman and Beth Sundheim. Message understanding conference - 6: A brief history. In *Proceedings of the International Conference on Computational Linguistics*, 1996.
8. N. Hsu and M. Dung. Generating finite-state transducers for semistructured data extraction from the web. *Journal of Information Systems, Special Issue on Semistructured Data*, 23(8), 1998.
9. C. Izard. *The face of emotion*. Appleton-Century-Crofts, 1971.
10. T. Joachims. A probabilistic analysis of the rocchio algorithm with tfidf for text categorization. In *Proceedings of the 14th International Conference on Machine Learning*, 1997.
11. T. Joachims. Text categorization with support vector machines. In *Proceedings of the European Conference on Machine Learning*, 1998.
12. Thorsten Joachims, Dayne Freitag, and Tom Mitchell. WebWatcher: A tour guide for the World Wide Web. In *Proceedings of the 15th International Joint Conference on Artificial Intelligence (IJCAI-97)*, pages 770–777, San Francisco, August23–29 1997. Morgan Kaufmann Publishers.
13. J. Konstantin, B. Miller, D. Maltz, J. Herlocker, L. Gordon, and J. Riedl. Grouplens: applying collaborative filtering to usenet news. *Communications of the ACM*, 40(3):77–87, 1997.
14. N. Kushmerick. Wrapper induction: efficiency and expressiveness. *Artificial Intelligence*, 118:15–68, 2000.
15. John Lafferty, Fernando Pereira, and Andrew McCallum. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the International Conference on Machine Learning*, 2001.
16. P. Maes. Agents that reduce work and information overload. *Communications of the ACM*, 37(7), 1994.
17. Andrew McCallum, Dayne Freitag, and Fernando Pereira. Maximum entropy Markov models for information extraction and segmentation. In *Proceedings of the Seventeenth International Conference on Machine Learning*, 2000.
18. A. Moukas. Amalthaea: Information discovery and filtering using a multiagent evolving ecosystem. In *Proceedings of the Conference on Practical Application of Intelligent Agents and Multi-Agent Technology*, 1996.
19. M Pazzani, J. Muramatsu, and D. Billsus. Syskill and webert: Identifying interesting web sites. In *Proceedings of the National Conference on Artificial Intelligence*, pages 54–61, 1996.

20. L. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–285, 1989.
21. T. Scheffer, C. Decomain, and S. Wrobel. Active hidden markov models for information extraction. In *Proceedings of the International Symposium on Intelligent Data Analysis*, 2001.
22. T. Scheffer and T. Joachims. Expected error analysis for model selection. In *Proceedings of the Sixteenth International Conference on Machine Learning*, 1999.
23. Tobias Scheffer and Stefan Wrobel. Active learning of partially hidden markov models. In *Proceedings of the ECML/PKDD Workshop on Instance Selection*, 2001.
24. M. Sehami, M. Craven, T. Joachims, and A. McCallum, editors. *Learning for Text Categorization, Proceedings of the ICML/AAAI Workshop.* AAAI Press, 1998.
25. Kristie Seymore, Andrew McCallum, and Roni Rosenfeld. Learning hidden markov model structure for information extraction. In *AAAI'99 Workshop on Machine Learning for Information Extraction*, 1999.
26. B. Sheth. Newt: A learning approach to personalized information filtering. Master's thesis, Department of Electiric Engineering and Computer Science, MIT, 1994.