A Comparison of Sentiment Analysis Techniques: Polarizing Movie Blogs

Michelle Annett and Grzegorz Kondrak

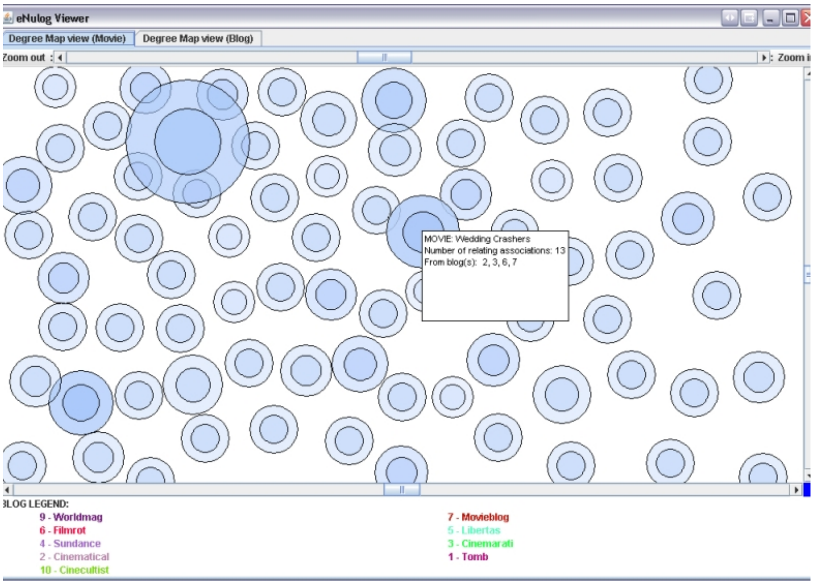
Department of Computing Science, University of Alberta
{mkannett,kondrak}@cs.ualberta.ca

Abstract. With the ever-growing popularity of online media such as blogs and social networking sites, the Internet is a valuable source of information for product and service reviews. Attempting to classify a subset of these documents using polarity metrics can be a daunting task. After a survey of previous research on sentiment polarity, we propose a novel approach based on Support Vector Machines. We compare our method to previously proposed lexical-based and machine learning (ML) approaches by applying it to a publicly available set of movie reviews. Our algorithm will be integrated within a blog visualization tool.

1 Introduction

Imagine the following scenario: you hear about a movie which opened in theaters last weekend that made \$75 million, but you have not heard anything about it. Before trekking to the movie theater and potentially wasting your money, you want to determine if this new movie is worth seeing or not. While this may seem like a trivial problem, it raises an important question: What sources of information do you use to assist you with your decision? Roughly ten years ago, one would poll the opinions of their friends or listen to movie critics such as Siskel and Ebert. Nowadays, most individuals consult websites or blogs such as *Yahoo!Movies* or *RottenTomatoes.com* to obtain this same information. While it is undoubtedly true that useful information can be obtained from these sources, the methods that one uses to gather this information, such as navigating through an endless number of websites or using an unintuitive RSS interface, are time consuming and potentially frustrating.

In order to assist users in managing the overwhelming amount of information present within the movie blog domain, and to increase the navigability between numerous blog sites, Tirapat *et al.* [1] have created *eNulog*, a blog visualization tool. *eNulog* mines and dynamically displays large collections of data, such as movie blogs, in an appealing and intuitive manner. The visualizations are based on a node and cluster structure, with each node representing a specific movie and each cluster containing nodes that are related in some way (either by director, genre, or actor). The application is very interactive: by selecting a node, the user can view each post relating to the selected movie and watch movie clusters form within the visualization (Figure 1).



MOVIE: Wedding Crashers
Number of relating associations: 13
From blog(s): 2, 3, 6, 7

LOG LEGEND:

9 - Worldmag
6 - Filmrot
4 - Sundance
2 - Cinematical
10 - Cinecultist
7 - Movieblog
5 - Libertas
3 - Cinemarati
1 - Tomb

Fig. 1. *eNulog* Interface

Currently, *eNulog* is unable to create polarity generalizations for a collection of movie reviews. In order to find out if a collection's composition is positive or negative, a user has to click on a movie node and read all of its blog posts. The lack of generalization functionality in *eNulog* provided the motivation for our research on sentiment analysis. Sentiment analysis, or statement polarity classification, is concerned with determining the relative positivity or negativity of a document, web page or selection of text. A program that could reliably determine the polarity of a collection of blog postings would allow users to save time and alleviate frustration.

The task of sentiment analysis is difficult due to the variability and complexity of language expressions. One of the biggest challenges of this task is brought about by *thwarted* and *negated* expressions. A thwarted expression contains a number of words that have a polarity which is opposite to the polarity of the expression itself. For example: 'Johnny Depp was alright. The previous two pirate movies were unrealistic and boring. The plot was awful. However, the special effects made the third pirate movie excellent.' In this statement, the number of negative words incorrectly implies that the statement is negative, when in reality it is actually positive and supportive. On the other hand, a negated expression is composed of a negating word (such as 'not' or 'never') followed by a noun, adjective, adverb or verb. It is easy for a human reader to discern the polarity of a statement such as, 'this movie was neither enjoyable nor entertaining', but to an automated entity or program, this is not an easy task. Due to the presence of thwarted and negated expressions, the task of sentiment analysis is non-trivial.

In this paper, we propose a novel approach based on Support Vector Machines. We investigate variations of feature vectors involving different sizes of feature vectors, different feature representations, and different feature types. We compare our approach against previously proposed techniques, within a unified framework, utilizing the same data set.

The paper is structured as follows. In Section 2, we provide the reader with a review of the state of the art of the field. In Section 3, we describe the methods we used in our experiments. The results of the experiments are presented in Section 4. Section 5 describes how our results will be applied to the *eNulog* program. Finally, in Section 6, we leave the reader with some concluding thoughts and future work that could be undertaken.

2 Related Work

Typically, within sentiment analysis classification, there are two main avenues of research. We first review the lexical approaches, which focus on building successful dictionaries, and then review the machine learning approaches, which are primarily concerned with feature vectors. Almost all of the related works presented in this section have been developed and tested on corpora from the movie domain.

A lexical approach typically utilizes a dictionary or lexicon of pre-tagged words. Each word that is present in a text is compared against the dictionary. If a word is present in the dictionary, then its polarity value is added to the 'total polarity score' of the text. For example, if a match has been found with the word 'excellent', which is annotated in the dictionary as positive, then the total polarity score of the blog is increased. If the total polarity score of a text is positive, then that text is classified as positive, otherwise it is classified as negative. Although naive in nature, many variants of this lexical approach have been reported to perform better than chance [2,3,4].

Because the classification of a statement is dependent upon the scoring it receives, there is a large volume of work devoted to discovering which lexical information works best. As a starting point for the field, Hatzivassiloglou and Wiebe [5] demonstrated that the subjectivity of an evaluative sentence could be determined through the use of a hand-tagged lexicon comprised solely of adjectives. They report over 80% accuracy on single phrases. Extending this work, Kennedy and Inkpen [2] utilized the same methodology and hand-tagged adjective lexicon as Hatzivassiloglou and Wiebe, but they tested the paradigm on a dataset composed of movie reviews. They reported a much lower accuracy rate of about 62%. Moving away from the hand-tagged lexicons, Turney utilized an Internet search engine to determine the polarity of words that would be included in the lexicon [3]. Turney performed two AltaVista search engine queries: one with a target word conjoined with the word 'good', and a second with the target word conjoined with the word 'bad'. The polarity of the target word was determined by the search result that returned the most hits. This approach improved accuracy to 65%.

In other research, Kamps *et al.* [4] and Andreevskaia *et al.* [6] chose to use the WordNet database to determine the polarity of words. We compared a target word to two pivot words (usually ‘good’ and ‘bad’) to find the minimum path distance between the target word and the pivot words in the WordNet hierarchy. The minimum path distance was converted to an incremental score and this value was stored with the word in the dictionary. The reported accuracy level of this approach was 64% [6]. An alternative to the WordNet metric, proposed by Turney and Littman, was to compute the semantic orientation of a word [7]. By subtracting a word’s association strength to a set of negative words from its association strength to a set of positive words, Turney and Littman were able to achieve an accuracy rate of 82% using two different semantic orientation statistic metrics.

The other main avenue of research within this area has utilized supervised machine learning techniques. Within the machine learning approach, a series of feature vectors are chosen and a collection of tagged corpora are provided for training a *classifier*, which can then be applied to an untagged corpus of text. In a machine learning approach, the selection of features is crucial to the success rate of the classification. Most commonly, a variety of unigrams (single words from a document) or n-grams (two or more words from a document in sequential order) are chosen as feature vectors. Other proposed features include the number of positive words, number of negating words, and the length of a document. Support Vector Machines (SVMs) [8,9] and the Naive Bayes algorithm are the most commonly employed classification techniques. The reported classification accuracy ranges between 63% and 82%, but these results are dependent upon the features selected.

3 Methods

In this section we describe the methodology behind the two sets of experiments that we performed.

3.1 Lexical Methods

The basic paradigm of the lexical approach is outlined below:

1. Preprocess each blog post (i.e. remove punctuation, strip HTML tags).
2. Initialize the total blog polarity score: $s \leftarrow 0$.
3. Tokenize each blog post. For each token, check if it is present in a dictionary of General Inquirer [10] (+ Yahoo! [11]) words.
   (a) If token is present in dictionary,
      i. If token is positive, then $s \leftarrow s + w$.
      ii. If token is negative, then $s \leftarrow s - w$.
4. Look at total blog polarity score s,
   (a) If $s >$ threshold, then classify the blog post as positive.
   (b) If $s <$ threshold, then classify the blog post as negative.

We investigated five variants of the lexical approach.

1. In the baseline approach, the dictionary is limited to a fixed number of positively and negatively tagged words.
2. To determine the effects of stemming on the classification accuracy, we applied a stemmer to each of the words in the dictionary (thereby increasing the size of the dictionary), as well as to each token in the blog post.
3. Because the dictionary does not contain 'slang' words, a part-of-speech tagger was applied to the blogs in the development set to search for 'slang' adjectives and adverbs. If a new 'slang' word was found, we used Turney's method in [3] to classify the word and add it to the dictionary.
4. To counter-balance the thwarted expressions, the words in the dictionary were assigned weights using the word's minimum path distance from the pivot words in WordNet.
5. The final variant combined variants two, three and four with the baseline approach.

It should be noted that the value of w in the general lexical method paradigm was dependent upon the paradigm variant that was being used. In the first, second and third variants, the value of w was 1, and in the fourth and fifth variants, the value of w was equivalent to the token's minimum path distance from the pivot words in WordNet.

3.2 Machine Learning Methods

The basic paradigm for the creation of the feature vectors was as follows:

1. Apply a part of speech tagger to each blog post in the development set.
2. Collect all of the adjectives/adverbs which were present in each blog post.
3. Make a popular word set composed of the top N adjectives and adverbs.
4. Traverse all of the blogs in the experimental set to create the following features:
   (a) Number of positive words
   (b) Number of negative words
   (c) Number of negating words
   (d) Presence, absence or frequency of each word in the popular word set (N words)

For the generation of feature vectors, we re-used some of the methods outlined in the previous section. We investigated the following four feature sets:

1. A fixed number of the most frequently occurring popular words (or unigrams), each of which is assigned an integer value representing the frequency of the word in the blog post.
2. Same features as in #1, but with binary representations of the unigrams (instead of integer values), corresponding to the presence or absence of a word in the blog post.
3. Same features as in #1 plus three *aggregate* features: the number of positive, negative, and neutral words present in the blog post.
4. Same features as in #3, but with binary representations of the unigrams as in #2.

4 Evaluation

In this section, we describe the results of our experiments with the lexical and the machine learning approaches.

4.1 Resources

We used the following resources:

1. Cornell Movie Review Dataset of Tagged Blogs (1000 positive and 1000 negative) [12].
2. List of 2000 positive words and 2000 negative words from the General Inquirer lists of adjectives [10].
3. Yahoo! Web Search API [11].
4. Porter Stemmer [13].
5. WordNet Java API [14].
6. Stanford Log Linear POS Tagger built with the Penn Treebank tag set [15].
7. WEKA Machine Learning Java API (only used for machine learning) [16].
8. SVM-Light Machine Learning Implementation [17].

Before both experiments were performed, we randomly divided the Cornell Movie Review Dataset of Tagged Blogs into a 200 member development set and a 1800 member training set (to ensure that our baseline would be evenly distributed, 100 positive and 100 negative blogs were extracted from the Cornell set). As we needed to find the most popular words present in blog posts, the development set was set aside and mined for this purpose. The results reported in Section 4.2 were obtained using the entire set of 1800 blogs and the results reported in Section 4.3 used this same set, but also applied 10-fold cross validation.

4.2 Lexical Results

Table 1 shows the results of applying the five variants of the lexical approach described in Section 3.1 to our test set. Stemming does not help classification much; the increase in accuracy is negligible. It appears that the additional word matches made possible by stemming are counter-balanced by the loss of information represented by morphological endings. On the other hand, the use of WordNet has a positive influence; the addition of the WordNet word weightings increases the accuracy level by 10%.

The addition of new words also improved accuracy, but not as dramatically as the WordNet weight inclusion did (57.7% versus 60.4%). We believe that this occurred because a majority of the words that were classified using Yahoo! ended up as positive, which created an imbalance in the dictionary (originally the dictionary was composed of 50% positive words and 50% negative words). Because the polarity of the experimental set was split at 50-50, any large imbalance in the distribution of the dictionary results in the classification being skewed in one direction (as more words are counted as positive), thereby reducing the number of correct classifications.

Table 1. The Accuracy of Various Lexical Methods (%)

Approach	**Accuracy**
Baseline	50.0
Baseline + Stemming	50.2
Baseline + Yahoo! Words	57.7
Baseline + WordNet	60.4
Baseline + Stemming + WordNet + Yahoo! Words	55.7

Combining all three ideas lead to a surprising drop in accuracy. By simultaneously increasing the number of words in the dictionary, applying a stemming algorithm, and assigning a weight to each dictionary word, we substantially increased the size of the dictionary. The proportion of positive and negative token matches did not change, and we hypothesize that for each positive match that was found, it was equally likely to find a negative match, thereby creating a neutral net effect.

Given these results, it appears that the selection of words that are included in the dictionary is very important for the lexical approach. If the dictionary is too sparse or exhaustive, one risks the chance of over or under analyzing the results, leading to a decrease in performance. In accordance with previous findings, our results confirm that it is difficult to surpass the 65% accuracy level using a purely lexical approach.

4.3 Machine Learning Results

During the initial experimentation phase of the machine learning approach, we used the WEKA package [16] to obtain a general idea of which ML algorithms and methods would be best suited to classify the dataset we had. Our preliminary experiments indicated that the SVM, Naive Bayes and Alternating Decision Tree (ADTree) algorithms were the most accurate (with the SVM results being superior). We then decided to use the very popular SVM-Light package [17] to further examine the benefits of using an SVM approach. The methodology described in Section 3.2 was used to obtain feature vectors for each of the blogs in the test set and the results were converted into WEKA and SVM-Light formats. A number of popular words, or N, was first tested to determine the optimal number of features to utilize for this classification task.

Table 2 shows the results of testing a varying number of features in SVM-Light. From the results, it is fairly clear to see that the utilization of a small number of features such as 50 is ineffective. While the results for such a small number of features are better than those obtained using a lexical based approach, they pale in comparison to the effects of using 1000 to 2000 features. Due to the nature of the corpora we are dealing with, it makes sense that the utilization of larger feature vectors is beneficial. Classifying an unstructured document, such as an online blog, cannot be done by using a small number of features that have a low probability of being present in the blog to begin with. Within a different domain, this approach might be feasible, but due to the variability

Table 2. SVM-Light Accuracy Results for Varying Numbers of Features (%)

Number of Features	Accuracy
50	66.2
600	68.1
1000	77.1
1600	77.4
2000	77.4

between writing styles and the complexity and size of the English language, this approach is inadequate. The more features one introduces into the equation, the higher the probability of successful classification by an automated ML technique.

Table 3 shows the results of utilizing four different feature representations gathered from our test set, in three different machine learning algorithms. The unigram feature representations were the most effective across all algorithms. We believe that the disappointing performance of the combination of the aggregate and unigram features can be attributed to the fact that they represent two different classes. The information contained in the aggregate features is already present in the unigram features. Instead of helping a ML classifier, the addition of the aggregate features has a confounding effect.

The difference between utilizing a presence/absence or frequency representation for each feature is small (2% to 3%). We hypothesize that the reasons for this difference are the same as the reasons that WordNet weights were more effective than categorical +/ − 1 weights in the lexical approach. The weights eliminate the neutralizing effects of the negating sentences and thwarted expressions. By keeping track of how many times a word was found, some of the errors caused by these types of expressions are eliminated.

In comparing the results obtained from the three different algorithms, it is fairly obvious that even the best decision tree algorithm (ADTree) was the least effective. We believe that this decrease in performance is due to the nature of the decision tree algorithm and the fact that a fairly large tree is needed to handle all of the feature attributes that are present in the datasets. Due to the inherent size of the tree, an unclassified testing instance has to traverse through many prediction nodes until it reaches a leaf node. The longer the path an instance has to travel, the higher the likelihood that an incorrect prediction will be made,

Table 3. Machine Learning Accuracy Results (%)

Approach	SVM-Light	NaiveBayes	ADTree
Unigram Integer	77.4	77.1	69.3
Unigram Binary	77.0	75.5	69.3
Unigram Integer + Aggregate	68.2	77.3	67.4
Unigram Binary + Aggregate	65.4	77.5	67.4

thereby decreasing the classification performance on this task. The Naive Bayes approach performs quite well in all situations (greater than 75%). The unigram results classified by the SVM algorithm are on a similar level.

After initially running the SVM-Light experiments, we observed that the confusion matrix results were heavily skewed to the negative side, for all the feature representation vectors. To correct this imbalance, we tried introducing a threshold variable into the results. This threshold variable was set to -0.2, and whenever a blog was assigned a value greater than -0.2, it was classified as positive. Unfortunately, although this modification did succeed in evening out the confusion matrix, it did not increase the overall accuracy of the results.

The results from Tables 1 and 3 clearly indicate the superiority of machine learning approaches. Even the worst of the ML results is superior to the best of the lexical results. It seems that the lexical approaches rely too heavily on semantic information. As both the lexical and ML approaches demonstrated, the inclusion of any type of 'dictionary' information in an experiment does not automatically increase the method's performance. Even though the ML results are superior, one must not forget that in order for a ML approach to be successful, a large corpus of tagged training data must first be collected and annotated, and this can be a challenging and expensive task.

5 Application

The *eNulog* program is composed of two sections, the visualization program and the dataset. The visualization program reads in XML formatted files, extracts multiple dependencies from these files and then visualizes each of these dependencies. The *eNulog* dataset contains 6000 untagged blog posts that were mined from 10 very popular online movie blogs from June to December of 2006. The original intent of the *eNulog* application was to allow users to navigate and make quick judgments about large collections of data. Although initial conclusions can be obtained from the program (e.g. the distance between nodes indicates the relative similarity between movies in terms of their genres), it is very time consuming to read through a large number of blog posts about a certain movie to gather the gist of what the posters had to say.

As we have demonstrated with our SVM approach, it is quite easy to obtain an acceptable level of accuracy when classifying movie blog posts. By applying our SVM method to the *eNulog* dataset, we 'color classified' all of the blog posts in the *eNulog* data set according to their polarity values relative to the -0.2 threshold that was discovered in the ML results (Figure 2). There are three different node colors which are present in the visualization: the red nodes represent movies that have been classified as negative, the green nodes indicate movies that are considered positive, and a yellow color is used to indicate instances where there were too few blogs to make an accurate polarity judgment or instances where the classification of a movie was neutral because the total blog score was close to

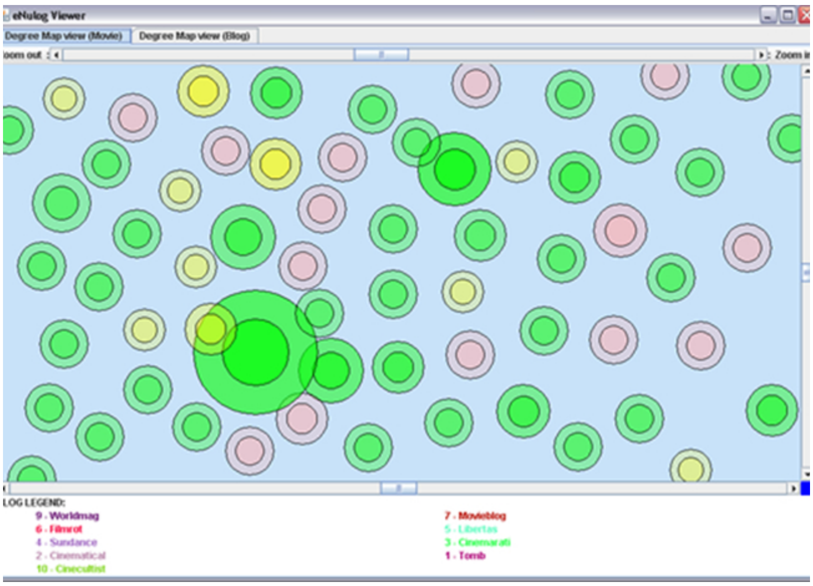


Fig. 2. *eNulog* with Sentiment Polarity Classification

zero. The addition of a 'polarity color coding' node scheme increases the utility of the *eNulog* program.

6 Conclusion

From the two experimental studies that were performed, we can conclude that the application of ML techniques for sentiment classification is quite successful. As expected, the ML results which were obtained supported the conclusion that the types of features which are chosen have a dramatic impact on the classification accuracy of an algorithm. As the lexical approaches have demonstrated, there is an upper bound on the accuracy level that a dictionary based approach can have, and it is currently unknown how this bound can be removed.

We foresee our future research on blog polarity proceeding in two main directions. First, we would like to experiment with classifying blog reviews according to a four or five star rating, as opposed to a simple binary classification scheme. Another, more challenging avenue of research would be to extend our application to take into account the user's attitudes and preferences toward specific genres, actors or actresses. Following Sato, Anse and Tabe [18], we could use *Kansei Engineering* to determine the relationship between one's movie tastes and certain features that are present in movies they have already seen. These relationships can then be exploited and extrapolated onto new, unseen movies, and our visualizations could transform the initial, general classifications into user-specific classifications, and in essence, turn *eNulog* into a recommender system.

References

1. Tirapat, T., Espiritu, C., Stroulia, E.: Taking the Community's Pulse, One Blog at a Time. In: Proceedings of the Sixth International Conference on Web Engineering, Palo Alto, CA, July 2006, pp. 169–176 (2006)
2. Kennedy, A., Inkpen, D.: Sentiment Classification of Movie and Product Reviews Using Contextual Valence Shifters. Computational Intelligence, 110–125 (2006)
3. Turney, P.: Thumbs Up or Thumbs Down? Semantic Orientation Applied to Unsupervised Classification of Reviews. In: Proceedings of ACL, Philadelphia, PA, July 2002, pp. 417–424 (2002)
4. Kamps, J., Marx, M., Mokken, R.J.: Using WordNet to Measure Semantic Orientation of Adjectives. In: LREC 2004, vol. IV, pp. 1115–1118 (2004)
5. Hatzivassiloglou, V., Wiebe, J.: Effects of Adjective Orientation and Gradability on Sentence Subjectivity. In: Proceedings of the 18th International Conference on Computational Linguistics, New Brunswick, NJ (2000)
6. Andreevskaia, A., Bergler, S., Urseanu, M.: All Blogs Are Not Made Equal: Exploring Genre Differences in Sentiment Tagging of Blogs. In: International Conference on Weblogs and Social Media (ICWSM-2007), Boulder, CO (2007)
7. Turney, P.D., Littman, M.L.: Measuring Praise and Criticism: Inference of Semantic Orientation from Association. ACM Transactions on Information Systems, 315–346 (2003)
8. Akshay, J.: A Framework for Modeling Influence, Opinions and Structure in Social Media. In: Proceedings of the Twenty-Second AAAI Conference on Artificial Intelligence, Vancouver, BC, July 2007, pp. 1933–1934 (2007)
9. Durant, K., Smith, M.: Mining Sentiment Classification from Political Web Logs. In: Proceedings of Workshop on Web Mining and Web Usage Analysis of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (WebKDD-2006), Philadelphia, PA (August 2006)
10. Stone, P.J., Dunphy, D.C., Smith, M.S.: The General Inquirer: A Computer Approach to Content Analysis. MIT Press, Cambridge (1966)
11. Yahoo! Search Web Services (October 2007), Online `http://developer.yahoo.com/search/`
12. Pang, B., Lee, L., Vaithyanathan, S.: Thumbs Up? Sentiment Classification Using Machine Learning Techniques. In: Proceedings of EMNLP-02, Association for Computational Linguistics, Philadelphia, PA, pp. 79–86 (2002)
13. No Title (October 2007), Online `http://tartarus.org/~martin/PorterStemmer/java.txt`
14. Fellbaum, C.: WordNet: An Electronic Lexical Database. Language, Speech, and Communication Series. MIT Press, Cambridge (1998)
15. Toutanova, K., Manning, C.D.: Enriching the Knowledge Sources Used in a Maximum Entropy Part-of-Speech Tagger. In: Proceedings of EMNLP/VLC-2000, Hong Kong, China, pp. 63–71 (2000)
16. Witten, H., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques, 2nd edn. Morgan Kaufmann, San Francisco (2005)
17. Joachims, T.: Making Large-Scale SVM Learning Practical. In: Schölkopf, B., Burges, C., Smola, A. (eds.) Advances in Kernel Methods - Support Vector Learning, pp. 169–184. MIT-Press, Cambridge (1999)
18. Sato, N., Anse, M., Tabe, T.: A Method for Constructing a Movie-Selection Support System Based on Kansei Engineering. In: Smith, M.J., Salvendy, G. (eds.) HCII 2007. LNCS, vol. 4557, pp. 526–534. Springer, Heidelberg (2007)