Automatically Assessing Review Helpfulness

Soo-Min Kim[†], Patrick Pantel[†], Timothy Chklovski[†], Marco Pennacchiotti[‡]

[†]Information Sciences Institute
University of Southern California
4676 Admiralty Way
Marina del Rey, CA 90292
{skim,pantel,timc}@isi.edu

[‡]ART Group - DISP
University of Rome "Tor Vergata"
Viale del Politecnico 1
Rome, Italy
pennacchiotti@info.uniroma2.it

Abstract

User-supplied reviews are widely and increasingly used to enhance e-commerce and other websites. Because reviews can be numerous and varying in quality, it is important to assess how *helpful* each review is. While review helpfulness is currently assessed manually, in this paper we consider the task of automatically assessing it. Experiments using SVM regression on a variety of features over Amazon.com product reviews show promising results, with rank correlations of up to 0.66. We found that the most useful features include the length of the review, its unigrams, and its product rating.

1 Introduction

Unbiased user-supplied reviews are solicited ubiquitously by online retailers like Amazon.com, Overstock.com, Apple.com and Epinions.com, movie sites like imdb.com, traveling sites like citysearch.com, open source software distributors like cpanratings.perl.org, and countless others. Because reviews can be numerous and varying in quality, it is important to rank them to enhance customer experience.

In contrast with ranking search results, assessing *relevance* when ranking reviews is of little importance because reviews are directly associated with the relevant product or service. Instead, a key challenge when ranking reviews is to determine which reviews the customers will find helpful.

Most websites currently rank reviews by their recency or product rating (e.g., number of *stars* in Amazon.com reviews). Recently, more sophisticated ranking schemes measure reviews by their *helpfulness*, which is typically estimated by having users manually assess it. For example, on Amazon.com, an interface allows customers to vote whether a particular review is helpful or not. Unfortunately, newly written reviews and reviews with few votes cannot be ranked as several assessments are required in order to properly estimate helpfulness. For example, for all MP3 player products on Amazon.com, 38% of the 20,919 reviews received three or fewer helpfulness votes. Another problem is that low-traffic items may never gather enough votes. Among the MP3 player reviews that were authored at least three months ago on Amazon.com, still only 31% had three or fewer helpfulness votes.

It would be useful to assess review helpfulness automatically, as soon as the review is written. This would accelerate determining a review's ranking and allow a website to provide rapid feedback to review authors.

In this paper, we investigate the task of automatically predicting review helpfulness using a machine learning approach. Our main contributions are:

- **A system for automatically ranking reviews according to *helpfulness***; using state of the art SVM regression, we empirically evaluate our system on a real world dataset collected from Amazon.com on the task of reconstructing the helpfulness ranking; and
- **An analysis of different classes of features most important to capture review helpfulness**; including *structural* (e.g., html tags, punctuation, review length), *lexical* (e.g., *n*-grams), *syntactic* (e.g., percentage of verbs and nouns), *semantic* (e.g., product feature mentions), and *meta-data* (e.g., star rating).

2 Relevant Work

The task of automatically assessing product review helpfulness is related to these broader areas

of research: automatic analysis of product reviews, opinion and sentiment analysis, and text classification.

In the thriving area of research on automatic analysis and processing of product reviews (Hu and Liu 2004; Turney 2002; Pang and Lee 2005), little attention has been paid to the important task studied here – assessing review helpfulness. Pang and Lee (2005) have studied prediction of product ratings, which may be particularly relevant due to the correlation we find between product rating and the helpfulness of the review (discussed in Section 5). However, a user's overall rating for the product is often already available. Helpfulness, on the other hand, is valuable to assess because it is not explicitly known in current approaches until many users vote on the helpfulness of a review.

In opinion and sentiment analysis, the focus is on distinguishing between statements of fact vs. opinion, and on detecting the polarity of sentiments being expressed. Many researchers have worked in various facets of opinion analysis. Pang et al. (2002) and Turney (2002) classified sentiment polarity of reviews at the document level. Wiebe et al. (1999) classified sentence level subjectivity using syntactic classes such as adjectives, pronouns and modal verbs as features. Riloff and Wiebe (2003) extracted subjective expressions from sentences using a bootstrapping pattern learning process. Yu and Hatzivassiloglou (2003) identified the polarity of opinion sentences using semantically oriented words. These techniques were applied and examined in different domains, such as customer reviews (Hu and Liu 2004) and news articles (TREC novelty track 2003 and 2004).

In text classification, systems typically use bag-of-words models, although there is some evidence of benefits when introducing relevant semantic knowledge (Gabrilovich and Markovitch, 2005). In this paper, we explore the use of some semantic features for review helpfulness ranking. Another potential relevant classification task is academic and commercial efforts on detecting email spam messages[1], which aim to capture a much broader notion of helpfulness. For an SVM-based approach, see (Drucker et al 1999).

Finally, a related area is work on automatic essay scoring, which seeks to rate the quality of an essay (Attali and Burstein 2006; Burstein et al. 2004). The task is important for reducing the human effort required in scoring large numbers of student essays regularly written for standard tests such as the GRE. The exact scoring approaches developed in commercial systems are often not disclosed. However, more recent work on one of the major systems, e-rater 2.0, has focused on systematizing and simplifying the set of features used (Attali and Burstein 2006). Our choice of features to test was partially influenced by the features discussed by Attali and Burstein. At the same time, due to differences in the tasks, we did not use features aimed at assessing essay structure such as discourse structure analysis features. Our observations suggest that even helpful reviews vary widely in their discourse structure. We present the features which we have used below, in Section 3.2.

[1] See http://www.ceas.cc/, http://spamconference.org/

3 Modeling Review Helpfulness

In this section, we formally define the learning task and we investigate several features for assessing review helpfulness.

3.1 Task Definition

Formally, given a set of reviews R for a particular product, our task is to rank the reviews according to their *helpfulness*. We define a review *helpfulness* function, h, as:

$$h(r \in R) = \frac{rating_{+}(r)}{rating_{+}(r) + rating_{-}(r)} \qquad (1)$$

where $rating_{+}(r)$ is the number of people that will find a review helpful and $rating_{-}(r)$ is the number of people that will find the review unhelpful. For evaluation, we resort to estimates of h from manual review assessments on websites like Amazon.com, as described in Section 4.

3.2 Features

One aim of this paper is to investigate how well different classes of features capture the helpfulness of a review. We experimented with various features organized in five classes: *Structural*, *Lexical*, *Syntactic*, *Semantic*, and *Meta-data*. Below we describe each feature class in turn.

Structural Features

Structural features are observations of the document structure and formatting. Properties such as review length and average sentence length are hypothesized to relate structural complexity to helpfulness. Also, HTML formatting tags could help in making a review more readable, and consequently more helpful. We experimented with the following features:

- *Length* (LEN): The total number of tokens in a syntactic analysis[2] of the review.
- *Sentential* (SEN): Observations of the sentences, including the number of sentences, the average sentence length, the percentage of question sentences, and the number of exclamation marks.
- *HTML* (HTM): Two features for the number of bold tags <b> and line breaks
.

Lexical Features

Lexical features capture the words observed in the reviews. We experimented with two sets of features:

- *Unigram* (UGR): The *tf-idf* statistic of each word occurring in a review.
- *Bigram* (BGR): The *tf-idf* statistic of each bigram occurring in a review.

For both unigrams and bigrams, we used lemmatized words from a syntactic analysis of the reviews and computed the *tf-idf* statistic (Salton and McGill 1983) using the following formula:

$$tf\ idf = \frac{tf \times \log(idf)}{N}$$

where N is the number of tokens in the review.

Syntactic Features

Syntactic features aim to capture the linguistic properties of the review. We grouped them into the following feature set:

- *Syntax* (SYN): Includes the percentage of parsed tokens that are open-class (i.e., nouns, verbs, adjectives and adverbs), the percentage of tokens that are nouns, the percentage of tokens that are verbs, the percentage of tokens that are verbs conjugated in the first person, and the percentage of tokens that are adjectives or adverbs.

Semantic Features

Most online reviews are fairly short; their sparsity suggests that bigram features will not perform well (which is supported by our experiments described in Section 5.3). Although semantic features have rarely been effective in many text classification problems (Moschitti and Basili 2004), there is reason here to hypothesize that a specialized vocabulary of *important* words might help with the sparsity. We hypothesized that good reviews will often contain: i) references to the features of a product (e.g., the *LCD* and *resolution* of a digital camera), and ii) mentions of sentiment words (i.e., words that express an opinion such as "*great* screen"). Below we describe two families of features that capture these semantic observations within the reviews:

- *Product-Feature* (PRF): The features of products that occur in the review, e.g., *capacity* of MP3 players and *zoom* of a digital camera. This feature counts the number of lexical matches that occur in the review for each product feature. There is no trivial way of obtaining a list of all the features of a product. In Section 5.1 we describe a method for automatically extracting product features from *Pro/Con* listings from Epinions.com. Our assumption is that pro/cons are the features that are important for customers (and hence should be part of a helpful review).
- *General-Inquirer* (GIW): Positive and negative sentiment words describing products or product features (e.g., "*amazing* sound quality" and "*weak* zoom"). The intuition is that reviews that analyze product features are more helpful than those that do not. We try to capture this analysis by extracting sentiment words using the publicly available list of positive and negative sentiment words from the General Inquirer Dictionaries[3].

Meta-Data Features

Unlike the previous four feature classes, meta-data features capture observations which are independent of the text (i.e., unrelated with linguistic features). We consider the following feature:

- *Stars* (STR): Most websites require reviewers to include an overall rating for the products that they review (e.g., star ratings in Amazon.com). This feature set includes the rating score (STR1) as well as the absolute value of the difference between the rating score and the average rating score given by all reviewers (STR2).

We differentiate meta-data features from semantic features since they require *external* knowledge that may not be available from certain review sites. Nowadays, however, most sites that collect user reviews also collect some form of product rating (e.g., Amazon.com, Overstock.com, and Apple.com).

[2] Reviews are analyzed using the Minipar dependency parser (Lin 1994).

[3] http://www.wjh.harvard.edu/~inquirer/homecat.htm

Table 1. Sample of 4 out of 43 reviews for the *iPod Photo 20GB* product from Amazon.com along with their ratings as well as their helpfulness ranks (from both the gold standard from Amazon.com and the SVM prediction of our best performing system described in Section 5.2).

Review Title	*Helpful Votes*	*Unhelpful Votes*	Rank(h)	
			Gold Standard	SVM Prediction
"iPod Moves to All-color Line-up"	215	11	7	1
"iPod: It's NOT Music to My Ears"	11	13	25	30
"The best thing I ever bought"	22	32	26	27
"VERY disappointing"	1	18	40	40

4 Ranking System

In this paper, we estimate the *helpfulness* function in Equation 1 using user ratings extracted from Amazon.com, where $rating_+(r)$ is the number of unique users that rated the review r as helpful and $rating_-(r)$ is the number of unique users that rated r as unhelpful.

Reviews from Amazon.com form a gold standard labeled dataset of {*review*, h(*review*)} pairs that can be used to train a supervised machine learning algorithm. In this paper, we applied an SVM (Vapnik 1995) package on the features extracted from reviews to learn the function h.

Two natural options for learning helpfulness according to Equation 1 are SVM Regression and SVM Ranking (Joachims 2002). Though learning to rank according to *helpfulness* requires only SVM Ranking, the helpfulness function provides non-uniform differences between ranks in the training set. Also, in practice, many products have only one review, which can serve as training data for SVM Regression but not SVM Ranking. Furthermore, in large sites such as Amazon.com, when new reviews are written it is inefficient to re-rank all previously ranked reviews. We therefore choose SVM Regression in this paper. We describe the exact implementation in Section 5.1.

After the SVM is trained, for a given product and its set of reviews R, we rank the reviews of R in decreasing order of $h(r)$, $r \in R$.

Table 1 shows four sample reviews for the *iPod Photo 20GB* product from Amazon.com, their total number of helpful and unhelpful votes, as well as their rank according to the helpfulness score h from both the gold standard from Amazon.com and using the SVM prediction of our best performing system described in Section 5.2.

5 Experimental Results

We empirically evaluate our review model and ranking system, described in Section 3 and Section 4, by comparing the performance of various feature combinations on products mined from Amazon.com. Below, we describe our experimental setup, present our results, and analyze system performance.

5.1 Experimental Setup

We describe below the datasets that we extracted from Amazon.com, the implementation of our SVM system, and the method we used for extracting features of reviews.

Extraction and Preprocessing of Datasets

We focused our experiments on two products from Amazon.com: *MP3 Players* and *Digital Cameras*.

Using Amazon Web Services API, we collected reviews associated with all products in the *MP3 Players* and *Digital Cameras* categories. For *MP3 Players*, we collected 821 products and 33,016 reviews; for *Digital Cameras*, we collected 1,104 products and 26,189 reviews.

In most retailer websites like Amazon.com, duplicate reviews, which are quite frequent, skew statistics and can greatly affect a learning algorithm. Looking for exact string matches between reviews is not a sufficient filter since authors of duplicated reviews often make small changes to the reviews to avoid detection. We built a simple filter that compares the distribution of word bigrams across each pair of reviews. A pair is deemed a duplicate if more than 80% of their bigrams match.

Also, whole products can be duplicated. For different product versions, such as iPods that can come in black or white models, reviews on Amazon.com are duplicated between them. We filter

Table 2. Overview of filtered datasets extracted from Amazon.com.

	MP3 Players	*Digital Cameras*
Total Products	736	1066
Total Reviews	11,374	14,467
Average Reviews/Product	15.4	13.6
Min/MaxReviews/Product	1 / 375	1 / 168

out complete products where each of its reviews is detected as a duplicate of another product (i.e., only one iPod version is retained).

The filtering of duplicate products and duplicate reviews discarded 85 products and 12,097 reviews for *MP3 Players* and 38 products and 3,692 reviews for *Digital Cameras*.

In order to have accurate estimates for the helpfulness function in Equation 1, we filtered out any review that did not receive at least five user ratings (i.e., reviews where less than five users voted it as helpful or unhelpful are filtered out). This filtering was performed before duplicate detection and discarded 45.7% of the *MP3 Players* reviews and 32.7% of the *Digital Cameras* reviews.

Table 2 describes statistics for the final datasets after the filtering steps. 10% of products for both datasets were withheld as development corpora and the remaining 90% were randomly sorted into 10 sets for 10-fold cross validation.

SVM Regression

For our regression model, we deployed the state of the art SVM regression tool SVM^{light} (Joachims 1999). We tested on the development sets various kernels including linear, polynomial (degrees 2, 3, and 4), and radial basis function (RBF). The best performing kernel was RBF and we report only these results in this paper (performance was measured using Spearman's correlation coefficient, described in Section 5.2).

We tuned the RBF kernel parameters C (the penalty parameter) and γ (the kernel width hyperparameter) performing full grid search over the 110 combinations of exponentially spaced parameter pairs (C,γ) following (Hsu et al. 2003).

Feature Extraction

To extract the features described in Section 3.2, we preprocessed each review using the Minipar dependency parser (Lin 1994). We used the parser tokenization, sentence breaker, and syntactic categorizations to generate the *Length*, *Sentential*, *Unigram*, *Bigram*, and *Syntax* feature sets.

In order to count the occurrences of product features for the *Product-Feature* set, we developed an automatic way of mining references to product features from Epinions.com. On this website, user-generated product reviews include explicit lists of *pros* and *cons*, describing the best and worst aspects of a product. For example, for MP3 players, we found the pro "*belt clip*" and the con "*Useless FM tuner*". Our assumption is that the *pro/con* lists tend to contain references to the product features that are important to customers, and hence their occurrence in a review may correlate with review helpfulness. We filtered out all single-word entries which were infrequently seen (e.g., *hold*, *ever*). After splitting and filtering the *pro/con* lists, we were left with a total of 9,110 unique features for *MP3 Players* and 13,991 unique features for *Digital Cameras*.

The *Stars* feature set was created directly from the star ratings given by each author of an Amazon.com review.

For each feature measurement f, we applied the following standard transformation:

$$\ln(f+1)$$

and then scaled each feature between [0, 1] as suggested in (Hsu et al. 2003).

We experimented with various combinations of feature sets. Our results tables use the abbreviations presented in Section 3.2. For brevity, we report the combinations which contributed to our best performing system and those that help assess the power of the different feature classes in capturing helpfulness.

5.2 Ranking Performance

Evaluating the quality of a particular ranking is difficult since certain ranking intervals can be more important than others (e.g., top-10 versus bottom-10) We adopt the Spearman correlation coefficient ρ (Spearman 1904) since it is the most commonly used measure of correlation between two sets of ranked data points[4].

For each fold in our 10-fold cross-validation experiments, we trained our SVM system using 9 folds. For the remaining test fold, we ranked each product's reviews according to the SVM prediction (described in Section 4) and computed the ρ

[4] We used the version of Spearman's correlation coefficient that allows for ties in rankings. See Siegel and Castellan (1988) for more on alternate rank statistics such as Kendall's *tau*.

Table 3. Evaluation of the feature combinations that make up our best performing system (in bold), for ranking reviews of Amazon.com *MP3 Players* and *Digital Cameras* according to *helpfulness*.

Feature Combinations	*MP3 Players*		*Digital Cameras*	
	Spearman†	*Pearson*†	*Spearman*†	*Pearson*†
LEN	0.575 ± 0.037	0.391 ± 0.038	0.521 ± 0.029	0.357 ± 0.029
UGR	0.593 ± 0.036	0.398 ± 0.038	0.499 ± 0.025	0.328 ± 0.029
STR1	0.589 ± 0.034	0.326 ± 0.038	0.507 ± 0.029	0.266 ± 0.030
UGR+STR1	0.644 ± 0.033	0.436 ± 0.038	0.490 ± 0.032	0.324 ± 0.032
LEN+UGR	0.582 ± 0.036	0.401 ± 0.038	0.553 ± 0.028	0.394 ± 0.029
LEN+STR1	0.652 ± 0.033	0.470 ± 0.038	0.577 ± 0.029	0.423 ± 0.031
LEN+UGR+STR1	**0.656 ± 0.033**	**0.476 ± 0.038**	**0.595 ± 0.028**	**0.442 ± 0.031**

LEN=*Length*; UGR=*Unigram*; STR=*Stars*

†95% confidence bounds are calculated using 10-fold cross-validation.

correlation between the ranking and the gold standard ranking from the test fold[5].

Although our task definition is to learn review rankings according to *helpfulness*, as an intermediate step the SVM system learns to predict the absolute helpfulness score for each review. To test the correlation of this score against the gold standard, we computed the standard Pearson correlation coefficient.

Results show that the highest performing feature combination consisted of the *Length*, the *Unigram*, and the *Stars* feature sets. Table 3 reports the evaluation results for every combination of these features with 95% confidence bounds. Of the three features alone, neither was statistically more significant than the others. Examining each pair combination, only the combination of *length* with *stars* outperformed the others. Surprisingly, adding *unigram* features to this combination had little effect for the *MP3 Players*.

Given our list of features defined in Section 3.2, helpfulness of reviews is best captured with a combination of the *Length* and *Stars* features. Training an RBF-kernel SVM regression model does not necessarily make clear the exact relationship between input and output variables. To investigate this relationship between length and helpfulness, we inspected their Pearson correlation coefficient, which was 0.45. Users indeed tend to find short reviews less helpful than longer ones: out of the 5,247 reviews for *MP3 Players* that contained more than 1000 characters, the average gold standard helpfulness score was 82%; the 204 reviews with fewer than 100 characters had on average a score of 23%. The explicit product rating, such as *Stars* is also an indicator of review helpfulness, with a Pearson correlation coefficient of 0.48.

The low Pearson correlations of Table 3 compared to the Spearman correlations suggest that we can learn the ranking without perfectly learning the function itself. To investigate this, we tested the ability of SVM regression to recover the target helpfulness score, given the score itself as the only feature. The Spearman correlation for this test was a perfect 1.0. Interestingly, the Pearson correlation was only 0.798, suggesting that the RBF kernel does learn the *helpfulness* ranking without learning the function exactly.

5.3 Results Analysis

Table 3 shows only the feature combinations of our highest performing system. In Table 4, we report several other feature combinations to show why we selected certain features and what was the effect of our five feature classes presented in Section 3.2.

In the first block of six feature combinations in Table 4, we show that the *unigram* features outperform the *bigram* features, which seem to be suffering from the data sparsity of the short reviews. Also, *unigram* features seem to subsume the information carried in our semantic features *Product-Feature* (PRF) and *General-Inquirer* (GIW). Although both PRF and GIW perform well as standalone features, when combined with unigrams there is little performance difference (for *MP3 Players* we see a small but insignificant decrease in performance whereas for *Digital Cameras* we see a small but insignificant improvement). Recall that PRF and GIW are simply subsets of review words that are found to be *product features* or *sentiment words*. The learning algorithm seems to discover on its own which

[5] Recall that the gold standard is extracted directly from user helpfulness votes on Amazon.com (see Section 4).

Table 4. Performance evaluation of various feature combinations for ranking reviews of *MP3 Players* and *Digital Cameras* on Amazon.com according to *helpfulness*. The first six lines suggest that unigrams subsume the semantic features; the next two support the use of the raw counts of product ratings (stars) rather than the distance of this count from the average rating; the final six investigate the importance of auxiliary feature sets.

Feature Combinations	*MP3 Players*		*Digital Cameras*	
	Spearman†	*Pearson*†	*Spearman*†	*Pearson*†
UGR	0.593 ± 0.036	0.398 ± 0.038	0.499 ± 0.025	0.328 ± 0.029
BGR	0.499 ± 0.040	0.293 ± 0.038	0.434 ± 0.032	0.242 ± 0.029
PRF	0.591± 0.037	0.400 ± 0.039	0.527 ± 0.030	0.316 ± 0.028
GIW	0.571 ± 0.036	0.381 ± 0.038	0.524 ± 0.030	0.333 ± 0.028
UGR+PRF	0.570 ± 0.037	0.375 ± 0.038	0.546 ± 0.029	0.348 ± 0.028
UGR+GIW	0.554 ± 0.037	0.358 ± 0.038	0.568 ± 0.031	0.324 ± 0.029
STR1	0.589 ± 0.034	0.326 ± 0.038	0.507 ± 0.029	0.266 ± 0.030
STR2	0.556 ± 0.032	0.303 ± 0.038	0.504 ± 0.027	0.229 ± 0.027
LEN+UGR+STR1	**0.656 ± 0.033**	**0.476 ± 0.038**	0.595 ± 0.028	0.442 ± 0.031
LEN+UGR+STR1+SEN	0.653 ± 0.033	0.470 ± 0.038	**0.599 ± 0.028**	**0.448 ± 0.030**
LEN+UGR+STR1+HTM	0.640 ± 0.035	0.459 ± 0.039	0.594 ± 0.028	0.442 ± 0.031
LEN+UGR+STR1+SYN	0.645 ± 0.034	0.469 ± 0.039	0.595 ± 0.028	0.447 ± 0.030
LEN+UGR+STR1+SEN+HTM+SYN	0.631 ± 0.035	0.453 ± 0.039	0.600 ± 0.028	0.452 ± 0.030
LEN+UGR+STR1+SEN+HTM+SYN+PRF+GIW	0.601 ± 0.035	0.396 ± 0.038	0.604 ± 0.027	0.460 ± 0.030

LEN=*Length*; SEN=*Sentential*; HTM=*HTML*; UGR=*Unigram*; BGR=*Bigram*; SYN=*Syntax*; PRF=*Product-Feature*; GIW=*General-Inquirer*; STR=*Stars*

†95% confidence bounds are calculated using 10-fold cross-validation.

words are most important in a review and does not use additional knowledge about the meaning of the words (at least not the semantics contained in PRF and GIW).

We tested two different versions of the *Stars* feature: i) the number of star ratings, *STR1*; and ii) the difference between the star rating and the average rating of the review, *STR2*. The second block of feature combinations in Table 4 shows that neither is significantly better than the other so we chose *STR1* for our best performing system.

Our experiments also revealed that our *structural* features *Sentential* and *HTML*, as well as our syntactic features, *Syntax*, did not show any significant improvement in system performance. In the last block of feature combinations in Table 4, we report the performance of our best performing features (*Length*, *Unigram*, and *Stars*) along with these other features. Though none of the features cause a performance deterioration, neither of them significantly improves performance.

5.4 Discussion

In this section, we discuss the broader implications and potential impacts of our work, and possible connections with other research directions.

The usefulness of the *Stars* feature for determining review helpfulness suggests the need for developing automatic methods for assessing product ratings, e.g., (Pang and Lee 2005).

Our findings focus on predictors of helpfulness of reviews of tangible consumer products (consumer electronics). Helpfulness is also solicited and tracked for reviews of many other types of entities: restaurants (citysearch.com), films (imdb.com), reviews of open-source software modules (cpanratings.perl.org), and countless others. Our findings of the importance of *Length*, *Unigrams*, and *Stars* may provide the basis of comparison for assessing helpfulness of reviews of other entity types.

Our work represents an initial step in assessing helpfulness. In the future, we plan to investigate other possible indicators of helpfulness such as a reviewer's reputation, the use of comparatives (e.g., *more* and *better than*), and references to other products.

Taken further, this work may have interesting connections to work on personalization, social networks, and recommender systems, for instance by identifying the reviews that a particular user would find helpful.

Our work on helpfulness of reviews also has potential applications to work on automatic gen-

eration of review information, by providing a way to assess helpfulness of automatically generated reviews. Work on generation of reviews includes review summarization and extraction of useful reviews from blogs and other mixed texts.

6 Conclusions

Ranking reviews according to user *helpfulness* is an important problem for many online sites such as Amazon.com and Ebay.com. To date, most websites measure helpfulness by having users manually assess how helpful each review is to them. In this paper, we proposed an algorithm for automatically assessing helpfulness and ranking reviews according to it. Exploiting the multitude of user-rated reviews on Amazon.com, we trained an SVM regression system to learn a helpfulness function and then applied it to rank unlabeled reviews. Our best system achieved Spearman correlation coefficient scores of 0.656 and 0.604 against a gold standard for MP3 players and digital cameras.

We also performed a detailed analysis of different features to study the importance of several feature classes in capturing helpfulness. We found that the most useful features were the length of the review, its unigrams, and its product rating. Semantic features like mentions of product features and sentiment words seemed to be subsumed by the simple unigram features. Structural features (other than length) and syntactic features had no significant impact.

It is our hope through this work to shed some light onto what people find helpful in user-supplied reviews and, by automatically ranking them, to ultimately enhance user experience.

References

Attali, Y. and Burstein, J. 2006. Automated Essay Scoring With e-rater® V.2. *Journal of Technology, Learning, and Assessment*, *4*(3).

Burstein, J., Chodorow, M., and Leacock, C. 2004. Automated essay evaluation: the criterion online writing service. *AI Magazine.* 25(3), pp 27–36.

Drucker,H., Wu,D. and Vapnik,V. 1999. Support vector machines for spam categorization. *IEEE Trans. Neural Netw.*, 10, 1048–1054.

Gabrilovich, E. and Markovitch, S. 2005. Feature Generation for Text Categorization Using World Knowledge. In *Proceedings of IJCAI-2005*.

Hsu, C.-W.; Chang, C.-C.; and Lin, C.-J. 2003. A practical guide to SVM classification. *Technical report*, Department of Computer Science and Information Technology, National Taiwan University.

Hu, M. and Liu, B. 2004. *Mining and summarizing customer reviews.* KDD'04. pp.168 – 177

Kim, S. and Hovy, E. 2004. Determining the Sentiment of Opinions. *Proceedings of COLING-04.*

Joachims, T. 1999. Making Large-Scale SVM Learning Practical. In B. Schölkopf, C. Burges, and A. Smola (eds), *Advances in Kernel Methods: Support Vector Learning*. MIT Press. Cambridge, MA.

Joachims, T. 2002. Optimizing Search Engines Using Clickthrough Data. In Proceedings of ACM KDD-02.

Moschitti, A. and Basili R. 2004. Complex Linguistic Features for Text Classification: A Comprehensive Study. In *Proceedings of ECIR 2004*. Sunderland, U.K.

Pang, B, L. Lee, and S. Vaithyanathan. 2001. Thumbs up? Sentiment Classification using Machine Learning Techniques. *Proceedings of EMNLP 2002.*

Pang, B. and Lee, L. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the ACL*, 2005.

Riloff , E. and J. Wiebe. 2003. Learning Extraction Patterns for Subjective Expressions. In *Proc. of EMNLP-03*.

Riloff, E., J. Wiebe, and T. Wilson. 2003. Learning Subjective Nouns Using Extraction Pattern Bootstrapping. *Proceedings of CoNLL-03*

Rose, C., Roque, A., Bhembe, D., and Vanlehn, K. 2003. A Hybrid Text Classification Approach for Analysis of Student Essays. In *Proc. of the HLT-NAACL*, 2003.

Salton, G. and McGill, M. J. 1983. *Introduction to Modern Information Retrieval*. McGraw Hill.

Siegel, S. and Castellan, N.J. Jr. 1988. *Nonparametric Statistics for the Behavioral Sciences*. McGraw-Hill.

Spearman C. 1904. The Proof and Measurement of Association Between Two Things. *American Journal of Psychology*, 15:72–101.

Turney, P. 2002. Thumbs Up or Thumbs Down? Semantic Orientation Applied to Unsupervised Classification of Reviews. *Proceedings of the 40th Annual Meeting of the ACL*, Philadelphia, 417–424.

Vapnik, V.N. 1995. *The Nature of Statistical Learning Theory*. Springer.

Wiebe, J, R. Bruce, and T. O'Hara. 1999. Development and use of a gold standard data set for subjectivity classifications. *Proc. of the 37th Annual Meeting of the Association for Computational Linguistics(ACL-99)*, 246–253.

Yu, H. and Hatzivassiloglou, V. 2003. Towards Answering Opinion Questions: Separating Facts from Opinions and Identifying the Polarity of Opinion Sentences. *Proceedings of EMNLP 2003.*