Assigning Polarity Scores to Reviews Using Machine Learning Techniques

Daisuke Okanohara[1] and Jun'ichi Tsujii[1,2,3]

[1] Department of Computer Science, University of Tokyo,
Hongo, 7-3-1, Bunkyo-ku, Tokyo 113-0013
[2] CREST, JST, Honcho, 4-1-8, Kawaguchi-shi, Saitama 332-0012
[3] School of Informatics, University of Manchester,
POBox 88, Sackville St, Manchester, M60 1QD, UK
{hillbig, tsujii}@is.s.u-tokyo.ac.jp

Abstract. We propose a novel type of document classification task that quantifies how much a given document (review) appreciates the target object using not binary polarity (*good* or *bad*) but a continuous measure called *sentiment polarity score* (sp-score). An sp-score gives a very concise summary of a review and provides more information than binary classification. The difficulty of this task lies in the quantification of polarity. In this paper we use support vector regression (SVR) to tackle the problem. Experiments on book reviews with five-point scales show that SVR outperforms a multi-class classification method using support vector machines and the results are close to human performance.

1 Introduction

In recent years, discussion groups, online shops, and blog systems on the Internet have gained popularity and the number of documents, such as reviews, is growing dramatically. *Sentiment classification* refers to classifying reviews not by their topics but by the polarity of their sentiment (e.g, positive or negative). It is useful for recommendation systems, fine-grained information retrieval systems, and business applications that collect opinions about a commercial product.

Recently, sentiment classification has been actively studied and experimental results have shown that machine learning approaches perform well [13,11,10,20]. We argue, however, that we can estimate the polarity of a review more finely. For example, both reviews A and B in Table 1 would be classified simply as *positive* in binary classification. Obviously, this classification loses the information about the difference in the degree of polarity apparent in the review text.

We propose a novel type of document classification task where we evaluate reviews with scores like five stars. We call this score the *sentiment polarity score* (sp-score). If, for example, the range of the score is from one to five, we could give five to review A and four to review B. This task, namely, ordered multi-class classification, is considered as an extension of binary sentiment classification.

In this paper, we describe a machine learning method for this task. Our system uses support vector regression (SVR) [21] to determine the sp-scores of

R. Dale et al. (Eds.): IJCNLP 2005, LNAI 3651, pp. 314–325, 2005.

Table 1. Examples of book reviews

	Example of Review	binary	sp-score (1,...,5)
Review A	I believe this is very good and a "must read" I can't wait to read the next book in the series.	plus	5
Review B	This book is not so bad. You may find some interesting points in the book.	plus	4

Table 2. Corpus A: reviews for Harry Potter series book. Corpus B: reviews for all kinds of books. The column of *word* shows the average number of words in a review, and the column of *sentences* shows the average number of sentences in a review.

sp-score	Corpus A			Corpus B		
	review	words	sentences	review	words	sentences
1	330	160.0	9.1	250	91.9	5.1
2	330	196.0	11.0	250	105.2	5.2
3	330	169.1	9.2	250	118.6	6.0
4	330	150.2	8.6	250	123.2	6.1
5	330	153.8	8.9	250	124.8	6.1

reviews. This method enables us to annotate sp-scores for arbitrary reviews such as comments in bulletin board systems or blog systems. We explore several types of features beyond a bag-of-words to capture key phrases to determine sp-scores: n-grams and references (the words around the reviewed object).

We conducted experiments with book reviews from *amazon.com* each of which had a five-point scale rating along with text. We compared pairwise support vector machines (pSVMs) and SVR and found that SVR outperformed better than pSVMs by about 30% in terms of the squared error, which is close to human performance.

2 Related Work

Recent studies on sentiment classification focused on machine learning approaches. Pang [13] represents a review as a feature vector and estimates the polarity with SVM, which is almost the same method as those for topic classification [1]. This paper basically follows this work, but we extend this task to a multi-order classification task.

There have been many attempts to analyze reviews deeply to improve accuracy. Mullen [10] used features from various information sources such as references to the "work" or "artist", which were annotated by hand, and showed that these features have the potential to improve the accuracy. We use reference features, which are the words around the fixed review target word (book), while Mullen annotated the references by hand.

Turney [20] used *semantic orientation*, which measures the distance from phrases to "excellent" or "poor" by using search engine results and gives the word polarity. Kudo [8] developed decision stumps, which can capture substructures embedded in text (such as word-based dependency), and suggested that subtree features are important for opinion/modality classification.

Independently of and in parallel with our work, two other papers consider the degree of polarity for sentiment classification. Koppel [6] exploited a neutral class and applied a regression method as ours. Pang [12] applied a metric labeling method for the task. Our work is different from their works in several respects. We exploited square errors instead of precision for the evaluation and used five distinct scores in our experiments while Koppel used three and Pang used three/four distinct scores in their experiments.

3 Analyzing Reviews with Polarity Scores

In this section we present a novel task setting where we predict the degree of sentiment polarity of a review. We first present the definition of sp-scores and the task of assigning them to review documents. We then explain an evaluation data set. Using this data set, we examined the human performance for this task to clarify the difficulty of quantifying polarity.

3.1 Sentiment Polarity Scores

We extend the sentiment classification task to the more challenging task of assigning rating scores to reviews. We call this score the sp-score. Examples of sp-scores include *five-star* and *scores out of 100*. Let sp-scores take discrete values[1] in a closed interval $[min...max]$. The task is to assign correct sp-scores to unseen reviews as accurately as possible. Let $\hat{y}$ be the predicted sp-score and y be the sp-score assigned by the reviewer. We measure the performance of an estimator with the mean square error:

$$\frac{1}{n}\sum_{i=1}^{n}(\hat{y_i} - y_i)^2, \tag{1}$$

where $(x_1, y_1), ..., (x_n, y_n)$ is the test set of reviews. This measure gives a large penalty for large mistakes, while ordered multi-class classification gives equal penalties to any types of mistakes.

3.2 Evaluation Data

We used book reviews on *amazon.com* for evaluation data[2] [3]. Each review has stars assigned by the reviewer. The number of stars ranges from one to five:

[1] We could allow sp-scores to have continuous values. However, in this paper we assume sp-scores take only discrete values since the evaluation data set was annotated by only discrete values.

[2] *http://www.amazon.com*

[3] These data were gathered from google cache using google API.

one indicates the worst and five indicates the best. We converted the number of stars into sp-scores $\{1, 2, 3, 4, 5\}$ [4]. Although each review may include several paragraphs, we did not exploit paragraph information.

From these data, we made two data sets. The first was a set of reviews for books in the *Harry Potter* series (Corpus A). The second was a set of reviews for books of arbitrary kinds (Corpus B). It was easier to predict sp-scores for Corpus A than Corpus B because Corpus A books have a smaller vocabulary and each review was about twice as large. To create a data set with a uniform score distribution (the effect of skewed class distributions is out of the scope of this paper), we selected 330 reviews per sp-score for Corpus A and 280 reviews per sp-score for Corpus B [5]. Table 2 shows the number of words and sentences in the corpora. There is no significant difference in the average number of words/sentences among different sp-scores.

Table 3. Human performance of sp-score estimation. Test data: 100 reviews of Corpus A with 1,2,3,4,5 sp-score.

	Square error
Human 1	0.77
Human 2	0.79
Human average	0.78
cf. Random	3.20
All3	2.00

Table 4. Results of sp-score estimation: Human 1 (left) and Human 2 (right)

	Assigned					
	1	2	3	4	5	Total
Correct						
1	12	7	0	1	0	20
2	7	8	4	1	0	20
3	1	1	13	5	0	20
4	0	0	4	10	6	20
5	0	1	2	7	10	20
Total	20	17	23	24	16	100

	Assigned					
	1	2	3	4	5	total
Correct						
1	16	3	0	1	0	20
2	11	5	3	1	0	20
3	2	5	7	4	2	20
4	0	1	2	1	16	20
5	0	0	0	2	18	20
Total	29	14	12	9	36	100

3.3 Preliminary Experiments: Human Performance for Assigning Sp-scores

We treat the sp-scores assigned by the reviewers as correct answers. However, the content of a review and its sp-score may not be related. Moreover, sp-scores may vary depending on the reviewers. We examined the universality of the sp-score.

[4] One must be aware that different scales may reflect the different reactions than just scales as Keller indicated [17].

[5] We actually corrected 25000 reviews. However, we used only 2900 reviews since the number of reviews with 1 star is very small. We examined the effect of the number of training data is discussed in 5.3.

We asked two researchers of computational linguistics independently to assign an sp-score to each review from Corpus A. We first had them learn the relationship between reviews and sp-scores using 20 reviews. We then gave them 100 reviews with uniform sp-score distribution as test data. Table 3 shows the results in terms of the square error. The *Random* row shows the performance achieved by random assignment, and the *All3* row shows the performance achieved by assigning 3 to all the reviews. These results suggest that sp-scores would be estimated with 0.78 square error from only the contents of reviews.

Table 4 shows the distribution of the estimated sp-scores and correct sp-scores. In the table we can observe the difficulty of this task: the precise quantification of sp-scores. For example, human B tended to overestimate the sp-score as 1 or 5. We should note that if we consider this task as binary classification by treating the reviews whose sp-scores are 4 and 5 as positive examples and those with 1 and 2 as negative examples (ignoring the reviews whose sp-scores are 3), the classification precisions by humans A and B are 95% and 96% respectively.

4 Assigning Sp-scores to Reviews

This section describes a machine learning approach to predict the sp-scores of review documents. Our method consists of the following two steps: extraction of feature vectors from reviews and estimation of sp-scores by the feature vectors. The first step basically uses existing techniques for document classification. On the other hand, the prediction of sp-scores is different from previous studies because we consider ordered multi-class classification, that is, each sp-score has its own class and the classes are ordered. Unlike usual multi-class classification, large mistakes in terms of the order should have large penalties. In this paper, we discuss two methods of estimating sp-scores: pSVMs and SVR.

4.1 Review Representation

We represent a review as a feature vector. Although this representation ignores the syntactic structure, word positions, and the order of words, it is known to work reasonably well for many tasks such as information retrieval and document classification. We use *binary*, *tf*, and *tf-idf* as feature weighting methods [15]. The feature vectors are normalized to have L^2 norm 1.

4.2 Support Vector Regression

Support vector regression (SVR) is a method of regression that shares the underlying idea with SVM [3,16]. SVR predicts the sp-score of a review by the following regression:

$$f : R^n \mapsto R, y = f(x) = \langle \boldsymbol{w} \cdot \boldsymbol{x} \rangle + b. \tag{2}$$

SVR uses an *ϵ-insensitive loss function*. This loss function means that all errors inside the ϵ cube are ignored. This allows SVR to use few support vectors and gives generalization ability. Given a training set, $(\boldsymbol{x_1}, y_1), ..., (\boldsymbol{x_n}, y_n)$, parameters $\boldsymbol{w}$ and b are determined by:

$$\begin{aligned} &\text{minimize } \tfrac{1}{2}\langle \boldsymbol{w} \cdot \boldsymbol{w} \rangle + C \textstyle\sum_{i=1}^{n} (\xi_i + \xi_i^*) \\ &\text{subject to} \quad (\langle \boldsymbol{w} \cdot \boldsymbol{x_i} \rangle + b) - y_i \leq \epsilon + \xi_i \\ &\qquad\qquad y_i - (\langle \boldsymbol{w} \cdot \boldsymbol{x_i} \rangle + b) \leq \epsilon + \xi_i^* \\ &\xi_i^{(*)} \geq 0 \qquad i = 1, ..., n. \end{aligned} \tag{3}$$

The factor $C > 0$ is a parameter that controls the trade-off between training error minimization and margin maximization. The loss in training data increases as C becomes smaller, and the generalization is lost as C becomes larger. Moreover, we can apply a kernel-trick to SVR as in the case with SVMs by using a kernel function.

This approach captures the order of classes and does not suffer from data sparseness. We could use conventional linear regression instead of SVR [4]. But we use SVR because it can exploit the kernel-trick and avoid over-training. Another good characteristic of SVR is that we can identify the features contributing to determining the sp-scores by examining the coefficients ($\boldsymbol{w}$ in (2)), while pSVMs does not give such information because multiple classifiers are involved in determining final results. A problem in this approach is that SVR cannot learn non-linear regression. For example, when given training data are $(x = 1, y = 1), (x = 2, y = 2), (x = 3, y = 8)$, SVR cannot perform regression correctly without adjusting the feature values.

4.3 Pairwise Support Vector Machines

We apply a multi-class classification approach to estimating sp-scores. pSVMs [7] considers each sp-score as a unique class and ignores the order among the classes. Given reviews with sp-scores $\{1, 2, .., m\}$, we construct $m \cdot (m-1)/2$ SVM classifiers for all the pairs of the possible values of sp-scores. The classifier for a sp-score pair $(avsb)$ assigns the sp-score to a review with a or b. The class label of a document is determined by majority voting of the classifiers. Ties in the voting are broken by choosing the class that is closest to the neutral sp-score (i.e, $(1 + m)/2$).

This approach ignores the fact that sp-scores are ordered, which causes the following two problems. First, it allows large mistakes. Second, when the number of possible values of the sp-score is large (e.g, $n > 100$), this approach suffers from the data sparseness problem. Because pSVMs cannot employ examples that have close sp-scores (e.g, sp-score = 50) for the classification of other sp-scores (e.g, the classifier for a sp-score pair $(51vs100)$).

4.4 Features Beyond Bag-of-Words

Previous studies [9,2] suggested that complex features do not work as expected because data become sparse when such features are used and a bag-of-words

Table 5. Feature list for experiments

Features	Description	Example in Fig.1 review 1
unigram	single word	(I) (believe) .. (series)
bigram	pair of two adjacent words	(I believe) ... (the series)
trigram	adjacent three words	(I believe this) ... (in the series)
inbook	words in a sentence including "book"	(I) (can't) ... (series)
aroundbook	words near "book" within two words.	(the) (next) (in) (the)

approach is enough to capture the information in most reviews. Nevertheless, we observed that reviews include many chunks of words such as "very good" or "must buy" that are useful for estimating the degree of polarity. We confirmed this observation by using n-grams. Since the words around the review target might be expected to influence the whole sp-score more than other words, we use these words as features. We call these features *reference*. We assume the review target is only the word "book", and we use "inbook" and "aroundbook" features. The "inbook" features are the words appear in the sentences which include the word "book". The "around book" are the words around the word "book" within two words. Table 5 summarizes the feature list for the experiments.

5 Experiments

We performed two series of experiments. First, we compared pSVMs and SVR. Second, we examined the performance of various features and weighting methods.

We used Corpus A/B introduced in Sec. 3.2 for experiment data. We removed all HTML tags and punctuation marks beforehand. We also applied the Porter stemming method [14] to the reviews.

We divided these data into ten disjoint subsets, maintaining the uniform class distribution. All the results reported below are the average of ten-fold cross-validation. In SVMs and SVR, we used *SVMlight*[6] with the quadratic polynomial kernel $K(x, z) = (\langle x \cdot z \rangle + 1)^2$ and set the control parameter C to 100 in all the experiments.

5.1 Comparison of pSVMs and SVR

We compared pSVMs and SVR to see differences in the properties of the regression approach compared with those of the classification approach. Both pSVMs and SVR used unigram/tf-idf to represent reviews. Table 6 shows the square error results for SVM, SVR and a simple regression (least square error) method for Corpus A/B. These results indicate that SVR outperformed SVM in terms of the square error and suggests that regression methods avoid large mistakes by taking account of the fact that sp-scores are ordered, while pSVMs does not. We also note that the result of a simple regression method is close to the result of SVR with a linear kernel.

[6] http://svmlight.joachims.org/

Table 6. Comparison of multi-class SVM and SVR. Both use unigram/tf-idf.

	Square error	
Methods	Corpus A	Corpus B
pSVMs	1.32	2.13
simple regression	1.05	1.49
SVR (linear kernel)	1.01	1.46
SVR (polynomial kernel $(\langle x \cdot z \rangle + 1)^2$)	**0.94**	**1.38**

Figure 1 shows the distribution of estimation results for humans (top left: human 1, top right: human 2), pSVMs (below left), and SVR (below right). The horizontal axis shows the estimated sp-scores and the vertical axis shows the correct sp-scores. Color density indicates the number of reviews. These figures suggest that pSVMs and SVR could capture the gradualness of sp-scores better than humans could. They also show that pSVMs cannot predict neutral sp-scores well, while SVR can do so well.

5.2 Comparison of Different Features

We compared the different features presented in Section 4.4 and feature weighting methods. First we compared different weighting methods. We used only unigram features for this comparison. We then compared different features. We used only tf-idf weighting methods for this comparison.

Table 7 summarizes the comparison results of different feature weighting methods. The results show that *tf-idf* performed well on both test corpora. We should note that simple representation methods, such as *binary* or *tf*, give comparable results to tf-idf, which indicates that we can add more complex features without considering the scale of feature values. For example, when we add word-based dependency features, we have some difficulty in adjusting these feature values to those of unigrams. But we could use these features together in binary weighting methods.

Table 8 summarizes the comparison results for different features. For Corpus A, *unigram + bigram* and *unigram + trigram* achieved high performance. The performance of *unigram + inbook* was not good, which is contrary to our intuition that the words that appear around the target object are more important than others. For Corpus B, the results was different, that is, n-gram features could not predict the sp-scores well. This is because the variety of words/phrases was much larger than in Corpus A and n-gram features may have suffered from the data sparseness problem. We should note that these feature settings are too simple, and we cannot accept the result of reference or target object (*inbook*/*aroundbook*) directly.

Note that the data used in the preliminary experiments described in Section 3.3 are a part of Corpus A. Therefore we can compare the results for humans with those for Corpus A in this experiment. The best result by the machine learning approach (0.89) was close to the human results (0.78).

To analyze the influence of n-gram features, we used the linear kernel $k(x, z) := \langle x \cdot z \rangle$ in SVR training. We used tf-idf as feature weighting. We then

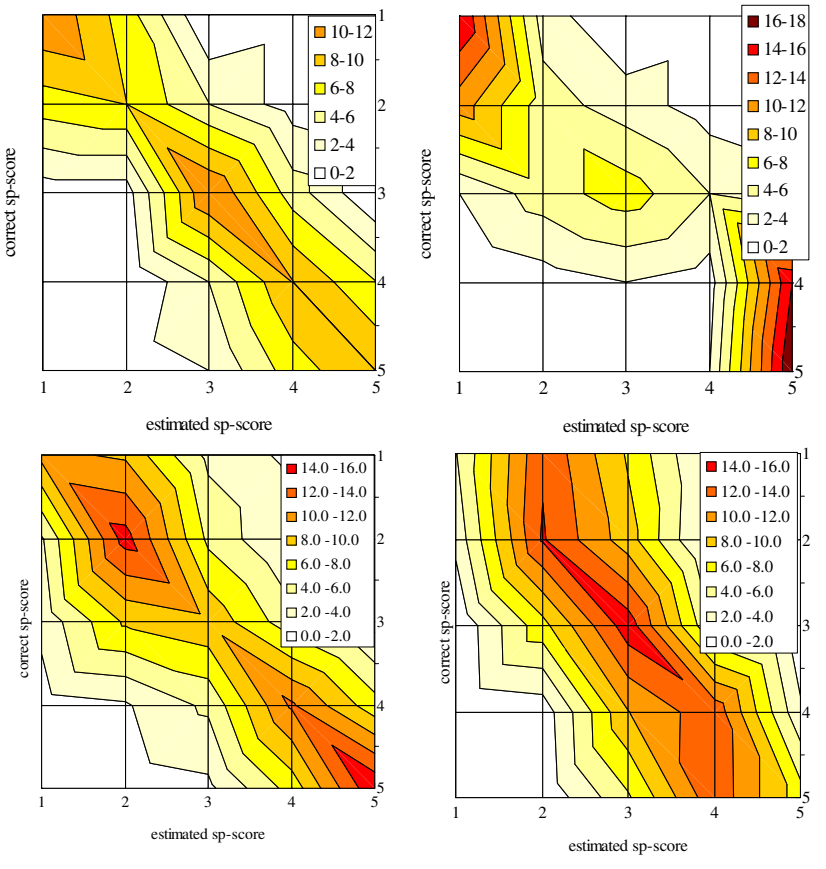

Fig. 1. Distribution of estimation results. Color density indicates the number of reviews. Top left: Human A, top right: Human B, below left: pSVMs, below right: SVR.

examined each coefficient of regression. Since we used the linear kernel, the coefficient value of SVR showed the polarity of a single feature, that is, this value expressed how much the occurrence of a feature affected the sp-score. Tables 9 shows the coefficients resulting from the training of SVR. These results show that neutral polarity words themselves, such as "all" and "age", will affect the overall sp-scores of reviews with other neutral polarity words, such as, "all ag (age)", "can't wait", "on (one) star", and "not interest".

5.3 Learning Curve

We generated learning curves to examine the effect of the size of training data on the performance. Figure 2 shows the results of a classification task using unigram */tf-idf* to represent reviews. The results suggest that the performance can still be improved by increasing the training data.

Table 7. Comparison results of different feature weighting methods. We used unigrams as features of reviews.

	Square error	
Weighting methods (unigram)	Corpus A	Corpus B
tf	1.03	1.49
tf-idf	**0.94**	**1.38**
binary	1.04	1.47

Table 8. Comparison results of different features. For comparison of different features we tf-idf as weighting methods.

	Square error	
Feature (tf-idf)	Corpus A	Corpus B
unigram (baseline)	0.94	1.38
unigram + bigram	**0.89**	1.41
unigram + trigram	0.90	1.42
unigram + inbook	0.97	**1.36**
unigram + aroundbook	0.93	1.37

Table 9. List of bigram features that have ten best/worst polarity values estimated by SVR in Corpus A/B. The column of *pol* expresses the estimated sp-score of a feature, i.e., only this feature is fired in a feature vector. (word stemming was applied)

Corpus A (best)		Corpus B (best)		Corpus A (worst)		Corpus B (worst)	
pol	bigram	pol	bigram	pol	bigram	pol	bigram
1.73	best book	1.64	the best	-1.61	at all	-1.19	veri disappoint
1.69	is a	1.60	read it	-1.50	wast of	-1.13	wast of
1.49	read it	1.37	a great	-1.38	potter book	-0.98	the worst
1.44	all ag	1.34	on of	-1.36	out of	-0.97	is veri
1.30	can't wait	1.31	fast food	-1.28	not interest	-0.96	! !
1.20	it is	1.22	harri potter	-1.18	on star	-0.85	i am
1.14	the sorcer's	1.19	highli recommend	-1.14	the worst	-0.81	the exampl
1.14	great !	1.14	an excel	-1.13	first four	-0.79	bui it
1.13	sorcer's stone	1.12	to read	-1.11	a wast	-0.76	veri littl
1.11	come out	1.01	in the	-1.08	no on	-0.74	onli to

6 Conclusion

In this paper, we described a novel task setting in which we predicted sp-scores - degree of polarity - of reviews. We proposed a machine learning method using SVR to predict sp-scores.

We compared two methods for estimating sp-scores: pSVMs and SVR. Experimental results with book reviews showed that SVR performed better in terms of the square error than pSVMs by about 30%. This result agrees with our

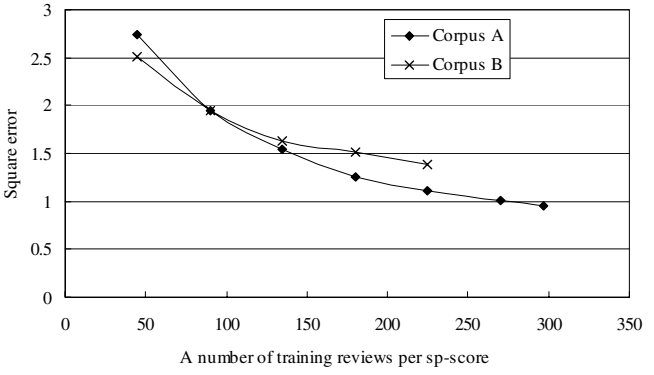

Fig. 2. Learning curve for our task setting for Corpus A and Corpus B. We used SVR as the classifier and unigram/*tf-idf* to represent of reviews.

intuition that pSVMs does not consider the order of sp-scores, while SVR captures the order of sp-scores and avoids high penalty mistakes. With SVR, sp-scores can be estimated with a square error of 0.89, which is very close to the square error achieved by human (0.78).

We examined the effectiveness of features beyond a bag-of-words and reference features (the words around the reviewed objects.) The results suggest that n-gram features and reference features contribute to improve the accuracy.

As the next step in our research, we plan to exploit parsing results such as predicate argument structures for detecting precise reference information. We will also capture other types of polarity than attitude, such as modality and writing position [8], and we will consider estimating these types of polarity.

We plan to develop a classifier specialized for ordered multi-class classification using recent studies on machine learning for structured output space [19,18] or ordinal regression [5] because our experiments suggest that both pSVMs and SVR have advantages and disadvantages. We will develop a more efficient classifier that outperforms pSVMs and SVR by combining these ideas.

References

1. T . Joachims. *Learning to Classify Text Using Support Vector Machines.* Kluwer, 2002.
2. C. Apte, F. Damerau, and S. Weiss. Automated learning of decision rules for text categorization. *Information Systems*, 12(3):233–251, 1994.
3. N. Cristianini and J. S. Taylor. *An Introduction to Support Vector Machines and other Kernel-based Learning Methods.* Cambridge University Press, 2000.
4. T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning.* Springer, 2001.
5. Ralf Herbrich, Thore Graepel, and Klaus Obermayer. Large margin rank boundaries for ordinal regression. In *Advances in Large Margin Classifiers*, pages 115–132. MIT press, 2000.

6. Moshe Koppel and Jonathan Schler. The importance of neutral examples for learning sentiment. In *In Workshop on the Analysis of Informal and Formal Information Exchange during Negotiations (FINEXIN)*, 2005.
7. U. Kresel. *Pairwise Classification and Support Vector Machines Methods.* MIT Press, 1999.
8. T. Kudo and Y. Matsumoto. A boosting algorithm for classification of semi-structured text. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 301–308, 2004.
9. D. Lewis. An evaluation of phrasal and clustered representations on a text categorization task. In *Proceedings of SIGIR-92, 15th ACM International Conference on Research and Development in Information Retrieval*, pages 37–50, 1992.
10. A. Mullen and N. Collier. Sentiment analysis using Support Vector Machines with diverse information sources. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL)*, 2004.
11. B. Pang and L. Lee. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL)*, pages 271–278, 2004.
12. B. Pang and L. Lee. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the 43nd Meeting of the Association for Computational Linguistics (ACL)*, 2005.
13. B. Pang, L. Lee, and S. Vaithyanathan. Thumbs up? sentiment classification using machine learning techniques. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 79–86, 2002.
14. M.F. Porter. An algorithm for suffix stripping, program. *Program*, 14(3):130–137, 1980.
15. F. Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1):1–47, 2002.
16. A. Smola and B. Sch. A tutorial on Support Vector Regression. Technical report, NeuroCOLT2 Technical Report NC2-TR-1998-030, 1998.
17. Antonella Sorace and Frank Keller. Gradience in linguistic data. *Lingua*, 115(11):1497–1524, 2005.
18. B. Taskar. *Learning Structured Prediction Models: A Large Margin Approach.* PhD thesis, Stanford University, 2004.
19. I. Tsochantaridis, T. Hofmann, T. Joachims, and Y. Altun. Support vector machine learning for interdependent and structured output spaces. In *Machine Learning, Proceedings of the Twenty-first International Conference (ICML)*, 2004.
20. P. D. Turney. Thumbs up or thumbs down? semantic orientation applied to unsupervised classification of reviews. In *Proceedings of the 40th Meeting of the Association for Computational Linguistics (ACL)*, pages 417–424, 2002.
21. V. Vapnik. *The Nature of Statistical Learning Theory.* Springer, 1995.