

Sentiment Classification with Interpolated Information Diffusion Kernels

Stephan Raaijmakers
TNO Information and Communication Technology
Delft
The Netherlands
stephan.raaijmakers@tno.nl

ABSTRACT

Information diffusion kernels - similarity metrics in non-Euclidean information spaces - have been found to produce state of the art results for document classification. In this paper, we present a novel approach to global sentiment classification using these kernels. We carry out a large array of experiments addressing the well-known movie review data set of Pang and Lee, a *de facto* benchmark, comparing information diffusion kernels with a standard RBF kernel machine. Our results show that interpolation of unigram and bigram information is beneficiary for sentiment classification.¹

1. INTRODUCTION

Given the huge growth of the blogosphere, and the increasing awareness by the industry of the merits of mining social media, sentiment mining –the automated analysis of sentiments and opinions– has firmly established itself on the map. Applications are diverse, ranging from mining customer feedback in blogs, web forums and e-mails, security investigations, and personalized advertising. This implies there are many approaches to mining sentiment. One can focus on generating alerts for negative opinions given a certain product or person, on exploring the process of opinion-formation around a certain topic (identifying opinion leaders), or on modeling the flow of sentiment in a certain textual domain, for instance. In this paper, we focus on binary sentiment classification: the labeling of entire texts as being positive or negative.

Global sentiment classification, the classification of entire documents as being (e.g.) positive, negative or neutral on average, can be seen to a large extent as a document classification problem. While sentiment seems to be a too intricate notion to capture in terms of crude categories such as posi-

tive, neutral or negative, this is not really different from any classification task, where class labels usually only partially cover the semantics of objects to be classified. The global sentiment of an entire document is built up from several opinions, usually expressed on (or even below) the sentence-level, and it is not a priori clear how to exactly compute the aggregate sum of sentiments for the entire document. Similarly, in document classification, the global topic of a text may be split up into several subtopics as well. Sentiment classification can be seen as a rough yet useful approximation of the sentiment in a document, ignoring the exact dynamics of sentiment flow, and its structural decomposition into linguistic units like sentences.

Our research question is the following. A substantial body of research has addressed the question which types of linguistic information are useful for sentiment classification, especially investigating the role of unigrams and bigrams. We will take up this issue, and investigate the division of labor between unigrams and bigrams for binary sentiment classification: classifying a document as either positive or negative. In doing so, we will also address the question how to represent texts containing sentiment: as binary vectors, or as term frequency vectors. We start off by presenting the ingredients for our experiments: data, kernel machines, and a hyperparameter estimation algorithm. We then describe our experiments, and draw conclusions.

2. DATA

We will focus on predicting the global sentiment for the annotated movie review dataset of [11], commonly referred to as the *polarity dataset*. This dataset, consisting of 1000 positive and 1000 negative movie reviews, has become a *de facto* benchmark dataset for global binary sentiment classification. In Table 1, two sample snippets from a positive and negative review are listed.

3. KERNELS

Our classification framework consists of *support vector machines* (SVM; [3]), classifiers that linearly separate data by (implicitly) projecting it into a high-dimensional space using *kernels*: similarity functions that compare data represented by feature vectors. We evaluate the use of information diffusion kernels, which measure similarity in non-Euclidean information spaces, and compare it to more standard RBF kernels.

Information diffusion kernels ([7]) are similarity functions that explicitly take into account geometric properties of data like curvature and angle. They have been found to produce

¹This work is supported by the European IST Programme Project FP6-0033812. This paper only reflects the authors' views and funding agencies are not liable for any use that may be made of the information contained herein.

Positive
kidman knows a star-making role when she sees it and she plays the conscience-deprived suzanne as the ultimate career driven , post-feminist uber-temptress from hell . she is ably abetted by wonderfully demented comic dialogue , which kidman delivers in a unselfconsciously funny way.
Negative
tons of illogical moments about ; more than i really have the inclination to list . and i don't mean illogical in the summer popcorn movie sense . that type of illogical can be fun . this film insults your intelligence more times than i care to remember . the last 20 minutes especially degenerates into such lunacy that you'll be laughing more than you'll be screaming .

Table 1: Sample snippets of positive and negative movie reviews.

state-of-the-art results for document classification, which is the reason why we evaluate them for the present problem.

Specifically, information diffusion kernels operate in curved L1- normalized information spaces with only a local Euclidian structure, and measure distance between datapoints along arcs connecting points. This contrasts with the standard vector space model of [17], where distance between points is Euclidean distance in high-dimensional spaces, irrespective of the geometry of these spaces. L1-normalized information spaces for textual objects emerge by simply normalizing word frequencies in documents represented by word sequences $w_1, \dots, w_n$:

$$L_1(w_1, \dots, w_n) = \frac{|w_1|}{\sum_i |w_i|}, \dots, \frac{|w_n|}{\sum_i |w_i|} \quad (1)$$

It is well-known that applying L1-normalization corresponds to embedding data into the multinomial manifold: an infinitely differentiable manifold that corresponds to the parameter space of $\mathbb{P}^n$ of the multinomial distribution. An infinitely differentiable manifold is called a *Riemannian manifold* when equipped with a distance metric measuring the distance between two arbitrary points. Riemannian manifolds are generalizations of Euclidean 2D-geometry to arbitrary dimensions.

In [7] and [8] it is shown that applying multinomial kernels to document classification tasks produces state-of-the-art results. In this work, the following kernel is proposed (x, y two feature vectors):

$$K_t^{ID}(x, y) = (4\pi t)^{-\frac{n}{2}} \exp \left(-\frac{1}{t} \arccos^2 \left(\sum_{i=1}^n \sqrt{x_i y_i} \right) \right) \quad (2)$$

This is a one-parameter kernel, n being the dimension of the data. In the context of support vector machines, for sufficiently small $t \in [0, \epsilon)$, this kernel is positive definite, guaranteeing a unique solution to the convex problem the kernel machine has to solve [7]. Interestingly, the so-called Bhattacharyya distance occurs as a subterm in (2):

$$B(p, q) = \sum_{x \in X} \sqrt{p(x)q(x)} \quad (3)$$

with p, q two probability distributions over the same event

space X . The Bhattacharyya distance is basically Euclidean distance restricted to probability distributions. In kernel (2), the inverse cosine ($\arccos$) is used to measure distance across curves, and the Bhattacharyya kernel introduces a local notion of Euclidean distance.

In [15] we proposed a two-parameter variant of (2):

$$K_{n,t}^{ID2}(x, y) = (4\pi t)^{\frac{n}{2}} \exp \left(-\frac{1}{t} \arccos^2 \left(\sum_{i=1}^m \sqrt{x_i y_i} \right) \right) \quad (4)$$

Here, n is a free parameter, part of a positive exponent, and no longer bound to the dimension of the data; the constant m is the original dimension of the data vectors. Kernel K_t^{ID} arises as a special case of $K_{n,t}^{ID2}$, by setting n to $-m$. In the experiments reported in [15] we observed clear positive effects on accuracy by adding n as a separate parameter and applied this kernel to a number of language learning tasks: classification problems on data not necessarily containing a document-like structure. For feature representations of such tasks, like part-of-speech tagging, or morphological analysis, we proposed a L1-embedding into the multinomial manifold based on class-feature value cooccurrence statistics, and demonstrated the usefulness of information diffusion kernels for this type of data. In the work reported here, we will use a variant of kernel (4), to be described below, which we will compare to a standard RBF kernel ([18]).

4. COMBINING LINGUISTIC INFORMATION FOR SENTIMENT MINING

Many types of information have been identified as relevant for sentiment classification during the last years: sentiment-bearing unigrams ([19]), word n -grams ([13],[6]), valency information of adjectives ([10]), valency-shifting properties of adverbs ([1],[6]), dependency relations ([10],[20]); [12] found that filtering out objective sentences is also helpful.

In this paper, we concentrate on using unigram and bigram information only, represented by frequency counts. These information sources are the natural ingredients for information diffusion kernels, once properly represented as normalized frequencies. RBF kernels, on the other hand, are frequently applied to either index vectors (where every feature flags the presence (yes/no) of a certain unigram of bigram), raw frequencies, or term weighting representations such as *tf.idf*.

Normalization of unigram and bigram frequencies will produce two separate multinomial probability distributions that need to be reconciled during the classification process. There are two options here. The first would be to view all unigram and bigrams strings as coming from the same distribution, ignoring intrinsic differences between the two distributions such as variance. Notice that during normalization the bigram sparseness may affect the unigram statistics.

The second option would be to interpolate the two information sources by assigning weights (Lagrange multipliers) to both unigram and bigram contributions, estimating these weights with an estimation procedure. Our hypothesis is that a good balance between these information sources will lead to accuracy gains. In the next section, we describe a general hyperparameter estimation procedure, needed independently to optimize our sentiment classifiers. Subsequently, we demonstrate how to use this algorithm to interpolate unigram and bigram information as part of this

optimization process.

4.1 Hyperparameter estimation

Classifiers are complex, parameterized decision functions. The parameters that influence the learning process are called hyperparameters. For SVM's, hyperparameters are the choice of kernel function, the regularization parameter C , and kernel parameters such as the degree of a polynomial kernel. Hyperparameters need to be accurately estimated; wrong settings can greatly influence results.

In [14], an elitist version of the well-known *cross-entropy (CE) method* for optimization was proposed (see the excellent tutorial of [5]). The CE algorithm is an iterative optimization procedure that performs parametrized sampling of a search space. It is particularly useful for sampling rare events: events that occur with low frequency, but which are useful from an objective point of view. Once the parameter vector along which the CE algorithm performs sampling is isomorphic to the type of solution to the optimization problem, the CE algorithm effectively becomes a search algorithm in a rare event space.

In [14] it was demonstrated that, using the notion of elitism from the field of genetic algorithms, it is possible to define a cross-entropy-style search algorithm for hyperparameter optimization of classifiers. This algorithm adapts, at a certain time tick t , a hyperparameter j in a hyperparameter vector $\hat{v}$ with the following update formula.

$$\hat{v}_{t,j} = \frac{\sum_{i=1}^n I_{\{S(X_i^t) \geq \gamma^t\}} W(X_i^t; E^t) X_{ij}^t}{\sum_{i=1}^n I_{\{S(X_i^t) \geq \gamma^t\}} W(X_i^t; E^t)} \quad (5)$$

Here, the $I_{\{S(X_i^t) \geq \gamma^t\}}$ term is a performance indicator function. Every X_i is a random solution drawn from the hyperparameter space, conditioned on the current hyperparameter vector and a width parameter μ . Every X_{ij} corresponds to a particular hyperparameter value, restricted to lie in the interval determined by the width parameter and the current solution derived by the algorithm:

$$X_{ij} \in \{\hat{v}_{t,j} * (1 - \mu), \hat{v}_{t,j} * (1 + \mu)\} \quad (6)$$

That is, every hyperparameter value X_{ij} in the candidate vector X_i is at most μ away from $\hat{v}_{t,j}$. The performance function $I()$ in 5 measures whether a certain X_i performs at least as good as the best score γ obtained so far, as measured by an independent objective function (like classifier accuracy). The $W()$ term embodies a *change of measure*: a steering mechanism to steer the search process into a certain 'good' direction. In our case, we steer search into the direction of the best ('elitist') result encountered during the whole optimization process, $E^t = \operatorname{argmax}_{\hat{v}_{i=1, \dots, t}} \gamma^i$. We measure (normalized) Euclidean distance between this best result and the candidate solution, and use this information (which lies in the interval $[0, 1]$) to steer search:

$$W(X_i^t; E^t) = 1 - \frac{\sqrt{\sum_{j=1}^m (X_{ij}^t - E_j^t)^2}}{\sqrt{\sum_{j=1}^m (X_{ij}^t)^2} \sqrt{\sum_{j=1}^m (E_j^t)^2}} \quad (7)$$

In [14], the effectiveness of this hyperparameter estimation algorithm was demonstrated for a number of datasets from different domains.

4.2 Kernel interpolation

The elitist hyperparameter estimation algorithm can be used to determine the optimal weights for unigram and bigram contributions to sentiment classification, once we make these weights hyperparameters of a kernel-based classifier, combining unigram and bigram information in a parameterizable way. While there are many ways to combine this information in one kernel, the following composed kernel combines them simply through a weighted sum. We factor out the contributions of both unigrams and bigrams by two separate Bhattacharyya subkernels. The weights λ_1 and λ_2 are interpolation weights that express the relative importance of the two information sources.

$$K_{n,t,\lambda_1,\lambda_2}^{ID2}(x, y, i, j) = (4\pi t)^{\frac{n}{2}} \exp \left(-\frac{1}{t} \arccos^2 \left(\frac{\lambda_1 \left[\sum_{k=1}^i \sqrt{x_i y_i} \right] + \lambda_2 \left[\sum_{i+1}^j \sqrt{x_i y_i} \right]}{2} \right) \right) \quad (8)$$

Every movie review is indexed for unigrams and bigrams. The result is a sparse feature vector, consisting of separately normalized frequencies of unigrams and bigrams. The two subscript parameters i and j indicate the highest index positions of the unigrams and bigrams in the vocabulary. This information is used by the kernel to factorize the unigram and bigram parts: any feature with an index higher than the maximum unigram index will be considered a bigram feature. By treating the interpolation weights as hyperparameters, we can use the elitist hyperparameter estimation algorithm to estimate these weights jointly with the 'normal' kernel hyperparameters n and t , which seems only natural to do. The *hyperkernel* (8) was implemented as an extension to the LIBSVM support vector machine toolkit ([4]).

5. RELATED WORK

[19] takes an unsupervised approach to polarity classification of documents, by inferring the semantic orientation of documents from the semantic orientation of phrases, which are gathered from the web by neighborhood queries containing positive or negative seed words. [13], using a predecessor of [11] consisting of 1400 documents, report negative effects on accuracy for polarity classification of documents when using bigrams as the sole features. They note a drop in accuracy of 5.8% for SVM's applied to the polarity dataset using only bigrams as features. Even the combination of unigrams and bigrams was outperformed by using unigrams alone. In [12], an accuracy of over 87% on the polarity data was achieved by focusing on subjective sentences only. [16] find that using both unigrams and bigrams, in combination with selected complex (syntactic) features taken from an information gain-based subsumption lattice of complex features produces the best results. They achieve 82.7% on the polarity dataset, but they report for 3-fold cross-validation only.

[6] obtain an accuracy of 85.9% on the polarity data (10CV), using SVM's and valency shifting bigrams, together with unigrams. [10] report 90.5% on the polarity data using additional information sources, such as valency information for adjectives, and filter out objective sentences first. To our knowledge, this is the best result reported ever on the polarity data. They report 89.2% using unigram, bigram and trigram features together, and also notice significant

drops in accuracy when using bigrams only. The latter is attributed by the authors to data sparseness: test documents frequently did not contain any of the bigrams in the training documents.

[9] also analyze the movie review data and investigate the combination of various information sources such as semantic orientation and emotive aspects of adjectives, the proximity of emotive phrases to the topic, and lemmatized unigrams. Using SVM's, they find interestingly that the best results (86% for 10-fold cross-validation for the movie data) are obtained by allocating information sources to separate SVM's, as opposed to merging them in one feature space and using a single classifier. In their approach, a hyperclassifier combines the model information of the subordinate classifiers, extracting features directly from the lower-level SVM models.

Work on local sentiment prediction, e.g. on the sentence level, is relevant to the work reported in this paper when identifying certain combinations of words that are important indicators for the sentiment of a document. [2] investigate the use of adverb-adjective combinations for local sentiment analysis. They propose an axiomatic system by which adverbs of degree modify the strength of sentiment of an adjective they combine with. This approach is related to [1] and [6], who investigate the use of valency shifting adverbs (like 'not'), that reverse the sentiment of adjectives in their scope. and the work of [20]), where polarity expressions are used for computing the polarity of their context. In sum, these lexical approaches, though addressing sentiment on a local level (like sentences), shed light on informative data sources beyond unigrams for sentiment analysis.

6. EXPERIMENTS

We set out to investigate the following questions. First, should we use all unigrams and bigrams in the training data, or a dedicated selection only? Secondly, should we interpolate unigram and bigram information, or just combine these two information sources, pretending they arise from the same distribution? Thirdly, should we use normalized frequency information, or just mere 'binary' presence (yes/no) of a certain vocabulary term? We arrive at the set of experimental conditions as listed in table 2, evaluated with the RBF kernel machine, the interpolated information diffusion kernel (8) and the non-interpolated information diffusion kernel (4). In order to apply the information diffusion kernels to the binary representations, we set all term frequencies to 1. We normalized these binary vectors by assigning every feature a value $1/n$, with n the number of terms from the vocabulary that were actually found in the review. For the interpolation experiments, we also investigated the effects of setting one of the weights λ_1 and λ_2 in kernel (8) to zero; setting $\lambda_1 = 0$ eliminates the unigram contribution, and $\lambda_2 = 0$ eliminates the bigram contribution.

6.1 Term selection

Following [10], we ranked all unigrams and bigrams t in the movie review data based on their weighted log-likelihood ratio (WLLR) scores, for the two classes $c \in \{+1, -1\}$:

$$WLLR(t, c) = P(t | c) \log \frac{P(t | c)}{P(t | -c)} \quad (9)$$

For both classes, we selected the 5000 highest ranked uni-

Interpolation	
E1	all terms, binary features
E2	all terms, normalized frequencies
E3	selected terms, binary features
E4	selected terms, normalized frequencies
No interpolation	
E5	all terms, binary features
E6	all terms, normalized frequencies
E7	selected terms, binary features
E8	selected terms, normalized frequencies
RBF	
E9	all terms, binary features
E10	all terms, normalized frequencies
E11	selected terms, binary features
E12	selected terms, normalized frequencies

Table 2: Experimental conditions.

Class	unigram
-1	bad, worst, stupid, boring, ridiculous, awful
+1	great, best, well, perfect, wonderful
Class	bigram
-1	this mess, the worst, worst movie a stupid, is terrible, waste of
+1	the best, is excellent, most powerful very effective, a great

Table 3: Highly WLLR-ranked unigrams and bigrams across the two classes.

grams and 5000 highest ranked bigrams from the training data as index vocabulary. Some illustrative unigrams and bigrams from the top 50 WLLR scores are listed in Table 3.

6.2 Experimental setup

For every experimental condition, we performed 10-fold cross-validation. We permuted the data, and split it into 10 training and test partitions. Every training partition T (90% of the data) was subsequently split into a secondary 90% training partition ($90\% \times 90\% = 81\%$ of the data), and a 10% development data partition. Finally, every development data partition was subsequently split into a 60% development training partition and a 40% development test partition. The two development data partitionings were used by the hyperparameter estimation algorithm to derive the hyperparameters for the kernel machine, for every separate fold. After the hyperparameter estimation algorithm found these hyperparameters, we trained on the 81% training data partitionings, and tested on the corresponding 10% test partitioning. Figure 1 illustrates the partitioning of a particular fold. Notice that we prudently kept the development data separate from the training data; this makes our results not exactly comparable to 10CV results reported for this data that do not employ development data.²

We measured significance of results using a paired t -test,

²In fact, comparison of cross-validation results produced by different classifiers on the same dataset should actually be done on exactly the same splits.

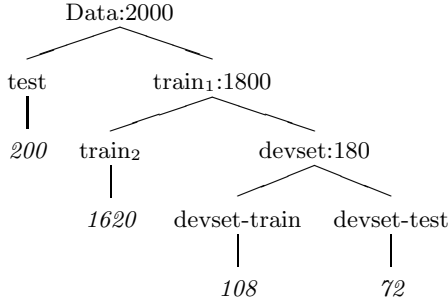


Figure 1: Sample data partitioning for 10CV. Actual training of the classifier took place on the ‘train₂’ partitions containing 1620 cases. Hyperparameter estimation took place on the ‘devset-train’ and ‘devset-test’ partitionings. The trained classifier was tested on the ‘test’ partitions containing 200 cases each.

Experimental condition	Interpolation		
	$\lambda_1 \neq 0, \lambda_2 \neq 0$	$\lambda_1 = 0$	$\lambda_2 = 0$
E1	87.5	81.9	86.7
E2	87.3	86.65	86.2
E3	88	81.85	86.7
E4	86.4	80.5	84
	No interpolation		
E5	82.65		
E6	80.5		
E7	86		
E8	86.8		
	RBF		
E9	86.8		
E10	84.25		
E11	85.8		
E12	81.1		

Table 4: Generalization accuracy (10CV), for the various experimental conditions (see table 2 for explanation of the labels).

at $p \leq 0.05$ and $p \leq .1$.

7. RESULTS

Table 4 list the results for the various experiments. Results for the interpolated kernel first of all underline the importance of the unigram contribution. The hyperparameter estimation algorithm predominantly assigned higher weights to the unigram subkernel across the various folds. The importance of unigrams becomes clear when manually deactivating the unigram subkernel, i.e. setting $\lambda_1 = 0$. The drops in accuracy are more dramatic than when eliminating the bigram contribution, sometimes even amounting to over 6%. This is in line with the findings of [13].

The combined result of using both unigram and bigram kernels, for the WLLR selected terms, with normalized binary vectors, produces the best results, showing that there is indeed useful, supplementary information in the bigrams. This classifier (E3) significantly outperformed the RBF baseline classifier (see table 5). Normalized binary vectors with

	E1	E3
E8	+ ($p \leq .1$)	+ ($p \leq .1$)
E9	+ ($p \leq .1$)	+ ($p \leq .1$)
E11	+ ($p \leq .05$)	+ ($p \leq .05$)

Table 5: Pairwise significance results, measured with a paired t-test.

WLLR-selected terms perform better³ than normalized frequency vectors.

[10] observed beneficiary effects of using the WLLR-ranked subset of unigrams and bigrams for a linear kernel with binary features. We observed that using all terms is beneficiary for the RBF kernel (classifier E9) only. For the interpolated kernel, the difference between WLLR selected terms and all terms vanishes: the difference between E1 and E3 is statistically significant only at $p \leq .35$, and the difference between E2 and E4 is significant only at $p \leq .42$.

Most importantly, our results specifically indicate that the combination of unigrams and bigrams yields better performance than using only one of these resources. Interpolation of unigrams and bigrams performs better than plain, unweighted combination. Further, RBF kernels are outperformed by information diffusion kernels, underlining the usability of these kernels.

8. CONCLUSIONS

In this paper, we investigated the use of unigram and bigram information for sentiment classification of movie reviews. We found that a composite kernel consisting of a separate unigram and bigram kernel, which were interpolated automatically with a cross-entropy-style hyperparameter estimation procedure, produced the best results. These results rank among the best reported using unigrams and bigrams on the movie review data set of [11], and are still open to improvement, by incorporating other sources of useful information reported in the literature, such as valency information of adjectives. We intend to expand this work into the direction of incorporating these sources of information by properly embedding them into L1-space, and weighting them with our hyperparameter estimation algorithm.

Being a general classification approach, kernel interpolation is amenable to multiclass classification of sentiment, and should be useful for predicting local sentiment on the sentence level as well. In subsequent work we will address subjectivity and objectivity prediction of sentences using this method.

Acknowledgements

I am indebted to Antal van den Bosch, my colleagues Wessel Kraaij and Franciska de Jong, and the anonymous reviewers for valuable feedback and suggestions.

9. REFERENCES

- [1] A. Andreevskaia, S. Bergler, and M. Urseanu. All blogs are not made equal, exploring genre differences in sentiment tagging of blogs. In *Proceedings of the International Conference on Weblogs and Social Media (ICWSM-2007) Boulder, CO, March 26-28*, pages 195–198, 2007.

³E3 is significantly better than E4 at $p \leq .13$.

- [2] F. Benamara, C. Cesarano, and D. Reforgiato. Sentiment analysis: Adjectives and adverbs are better than adjectives alone. In *Proceedings of the International Conference on Weblogs and Social Media (ICWSM-2007) Boulder, CO, March 26-28*, pages 203–206, 2007.
- [3] B. E. Boser, I. Guyon, and V. Vapnik. A training algorithm for optimal margin classifiers. In *Computational Learning Theory*, pages 144–152, 1992.
- [4] C.-C. Chang and C.-J. Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at <http://www.csie.ntu.edu.tw/~cjlin/libsvm>.
- [5] P. de Boer, D. Kroese, S. Mannor, and R. Rubinstein. A tutorial on the cross-entropy method. *Annals of Operations Research*, 2004.
- [6] Kennedy, Alistair, Inkpen, and Diana. Sentiment classification of movie reviews using contextual valence shifters. *Computational Intelligence*, 22(2):110–125, May 2006.
- [7] J. Lafferty and G. Lebanon. Diffusion kernels on statistical manifolds. *Journal of Machine Learning*, 6:129–163, 2005.
- [8] G. Lebanon. Sequential document representations and simplicial curves. In *Proceedings of the 22nd Conference on Uncertainty in Artificial Intelligence*, 2006.
- [9] T. Mullen and N. Collier. Sentiment analysis using support vector machines with diverse information sources. In *EMNLP*, pages 412–418, 2004.
- [10] V. Ng, S. Dasgupta, and S. M. N. Arifin. Examining the role of linguistic knowledge sources in the automatic identification and classification of reviews. In *ACL*, pages 611–618, 2006.
- [11] B. Pang and L. Lee. Movie review data, available from <http://www.cs.cornell.edu/people/pabo/movie-review-data>.
- [12] B. Pang and L. Lee. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the ACL*, pages 271–278, 2004.
- [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] S. Raaijmakers. Hyperparameter estimation with an elitist cross-entropy method. 2007. Technical report, TNO ICT, 2007. Available from <http://stephanraaijmakers.files.wordpress.com/2007/05/multinomial-languagelearning-v2.pdf>.
- [15] S. Raaijmakers. Language learning with information diffusion kernels. 2007. Technical report, TNO ICT, 2007, Available from <http://stephanraaijmakers.files.wordpress.com/2007/05/ce-hyperparam.pdf>.
- [16] E. Riloff, S. Patwardhan, and J. Wiebe. Feature Subsumption for Opinion Analysis. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 440–448, Sydney, Australia, July 2006.
- [17] G. Salton, A. Wong, and C. S. Yang. A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620, 1975.
- [18] B. Schoelkopf and A. J. Smola. *Learning with Kernels*. MIT Press, 2002.
- [19] P. Turney. Thumbs up or thumbs down? semantic orientation applied to unsupervised classification of reviews. In *ACL*, pages 417–424, 2002.
- [20] T. Wilson, J. Wiebe, and P. Hoffmann. Recognizing contextual polarity in phrase-level sentiment analysis. In *HLT05: Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 347–354, Morristown, NJ, USA, 2005. Association for Computational Linguistics.