A Language Modeling Approach to Sentiment Analysis

Yi Hu[1], Ruzhan Lu[1], Xuening Li[1,2], Yuquan Chen[1], and Jianyong Duan[1]

[1] Department of Computer Science and Engineering
Shanghai Jiao Tong University, Shanghai, China
[2] School of Foreign Studies
Southern Yangtze University, Wuxi, China
{huyi, lu-rz, xuening_li, yqchen, duan_jy}@cs.sjtu.edu.cn

Abstract. This paper presents a language modeling approach to the sentiment detection problem. It captures the subtle information in text processing to character the semantic orientation of documents as “thumb up” (positive) or “thumb down” (negative). To handle this problem, we propose an idea to estimate both the positive and negative language models from training collections. Tests are done through computing the Kullback-Leibler divergence between the language model estimated from test document and these two trained sentiment models. We assert the polarity of a test document by observing whether its language model is close to the trained “thumb up” model or the “thumb down” model. When compared with an outstanding classifier, i.e., SVMs on movie review corpus, language modeling approach showed its better performance.

Keywords: Sentiment Analysis; Language Modeling; KL-divergence.

1 Introduction

Traditional attention to document categorization lies in mapping a document to given topics such as sport, finance and politics [4]. Whereas, recent years have seen a growing interest in non-topical analysis, in which characterizations are sought of the opinions and feelings depicted in documents, rather than just their themes. The problem classifying a document as “thumb up” (positive) or “thumb down” (negative) is called sentiment classification. Labeling documents by such semantic orientations would provide succinct summaries and would be great useful in many intelligent information systems. Its immediate applications include mining webs and blocking junk mails.

Sentiment classification has become a hot direction for its broad applications, which has been attempted in different domains such as movie reviews, product reviews, and customer feedback reviews [1][2][6][8]. Some researchers have taken positive or negative word/phrase counting methods into account and determining if a word/phrase is positive or negative [9]. Other methods classify whole documents into positive and negative by employing machine-learning algorithms. Several learning algorithms are compared in [2] where it found that Support Vector Machines (SVMs) generally give better results. Their work shows that, generally, these algorithms are not able to achieve accuracies on sentiment classification comparable to those reported for standard topic-based categorization. The reason exists in many challenging aspects in this task. Intuitively, feelings in natural language are very often expressed in subtle and complex ways, which usually needs knowledge to deal with.

This paper presents a language modeling approach to analyze documents as positive or negative, which emphasizes on suitably estimating the “thumb up” and “thumb down” language models from training sets, and evaluating a test document represented with a language model via its distance from the two sentiment models. We also try SVMs, a powerful discriminative model, for this sentiment analysis.

The rest of the paper is organized as follows. Our method is formalized in Section 2. Section 3 follows with a description of preliminary experiments, and section 4 gives the conclusion. Note that this paper discusses an ongoing work and provides the framework of our idea and initial results, rather than a complete solution.

2 Language Modeling Approach to Sentiment Classification

In this section, we propose a language modeling approach to detecting semantic orientation in document. The motivation is very simple: the “thumb up” and “thumb down” languages are likely to be substantially different, i.e. they prefer to different language habits. We exploit this divergence in the language models to classify test document.

The “thumb up” orientation is represented with a positive language model θ_P that is a probability distribution over n-grams in positive collection. Accordingly, a negative language model θ_N represents the language model for “thumb down” orientation. A test document generates a language model θ_d. Note that a language model is a statistical model: probability distribution over language units, indicating the likelihood of observing these units in a language. Therefore a document can then compare its model with “thumb up” or “thumb down” model using distance mechanism:

$$\varphi(d;\theta_P,\theta_N) = Dis(\theta_d,\theta_P) - Dis(\theta_d,\theta_N) : \begin{cases} < 0 & \text{"}\textit{thumb up}\text{"} \\ > 0 & \text{"}\textit{thumb down}\text{"} \end{cases}. \tag{1}$$

Where *Dis(p,q)* is the distance between two distributions *p* and *q*. This formula expresses the classifying idea that if $Dis(\theta_d,\theta_P)$ is smaller than $Dis(\theta_d,\theta_N)$, it means the test document *d* is closer to “thumb up”. Otherwise, if $Dis(\theta_d,\theta_P)$ is greater than $Dis(\theta_d,\theta_N)$, “thumb down”. Note that if $\varphi(d;\theta_P,\theta_N)$ equals to zero, the test document is regarded as “neutral”, but this case has not been discussed in our work. Next subsection, we exploit the Kullback-Leibler Divergence as the distance measure.

2.1 Using Kullback-Leibler Divergence for Sentiment Classification

Given two probability mass functions $p(x)$ and $q(x)$, $D(p \| q)$, the Kullback-Leibler divergence (or relative entropy) between *p* and *q* is defined as

$$D(p \| q) = \sum_x p(x) \log\left(\frac{p(x)}{q(x)}\right). \tag{2}$$

It is easy to show that $D(p\,||\,q)$ is always non-negative and is zero if and only if $p = q$. Even though it is not a true distance between distributions because it is not symmetric and does not satisfy the triangle inequality, it is still often useful to think of the KL-divergence as a "distance" between distributions [3]. Formally, the KL-divergence between probability distributions θ_d and θ_P / θ_N is calculated by:

$$\begin{cases} D(\hat{\theta}_d \,||\, \hat{\theta}_P) = \sum_{n-gram} \Pr(n-gram \,|\, \hat{\theta}_d) \log\left(\frac{\Pr(n-gram \,|\, \hat{\theta}_d)}{\Pr(n-gram \,|\, \hat{\theta}_P)} \right) \\ D(\hat{\theta}_d \,||\, \hat{\theta}_N) = \sum_{n-gram} \Pr(n-gram \,|\, \hat{\theta}_d) \log\left(\frac{\Pr(n-gram \,|\, \hat{\theta}_d)}{\Pr(n-gram \,|\, \hat{\theta}_N)} \right) \end{cases} . \tag{3}$$

Where $\hat{\theta}$ is the estimated model for the real θ and $\Pr(n-gram \,|\, \hat{\theta})$ is the probability of n-gram given the estimated model $\hat{\theta}$.

Once we have a language models to represent test document and a score based on its distance to the two sentiment language models, we classify the test document as "thumb up" or "thumb down". Finally, substituting equation (3) into equation (1) we have a new sentiment classifying function:

$$\begin{aligned} \varphi(d; \hat{\theta}_P, \hat{\theta}_N) &= D(\hat{\theta}_d \,||\, \hat{\theta}_P) - D(\hat{\theta}_d \,||\, \hat{\theta}_N) \\ &= \sum_{n-gram} \Pr(n-gram \,|\, \hat{\theta}_d) \log\left(\frac{\Pr(n-gram \,|\, \hat{\theta}_N)}{\Pr(n-gram \,|\, \hat{\theta}_P)} \right) \end{aligned} . \tag{4}$$

In this study, we only employ word-based unigrams and bigrams as model parameters. Because of the data sparseness problem, higher order n-grams (n >= 3) have not been discussed, even if the higher order n-grams might approximate the true language model in theory. But it is possible to use n-grams of higher orders in the same framework. In general, the computation of the above formula involves a sum over all the n-grams that have a non-zero probability according to $\Pr(n-gram \,|\, \hat{\theta})$. However, when $\hat{\theta}$ is based on certain general smoothing technique, the computation would assign a non-zero probability to unseen n-gram according to $\Pr(n - gram \,|\, \hat{\theta}_{smooth})$. We also observe the smoothing effect in language modeling of sentiment.

2.2 Estimation for Model Parameters

Usually, the real language models (θ_P and θ_N) are unknown. They are estimated by training from two available collections labeled with "positive" and "negative" to obtain $\hat{\theta}_P$ and $\hat{\theta}_N$, respectively. $\hat{\theta}_d$ has the similar meaning.

For investigating the ability of language modeling approach, we use two methods to estimate the unigrams and bigrams distribution: <1> the Maximum Likelihood Estimate (MLE); <2> the smoothing estimation for these three language models.

MLE for Unigrams and Bigrams

MLE is used widely for model estimate, so we directly give the formula (5) and simply analyze it as follows.

$$\begin{cases} \Pr_{ML}(w_i \mid "s") = \dfrac{\#(w_i \; in \; "s")}{\#(* \; in \; "s")} & s \in \{d,P,N\} \quad for\ unigram \\ \Pr_{ML}(w_i \mid w_{i-1}, "s") = \dfrac{\#(w_{i-1} w_i \; in \; "s")}{\#(w_{i-1} * \; in \; "s")} & s \in \{d,P,N\} \quad for\ bigram \end{cases} . \tag{5}$$

What we have to explain in (5) is the "*s*", which can represent a test document (*d*), the "thumb up" collection (*P*) or the "thumb down" collection (*N*). The $\#(n-gram)$ denotes the n-gram occurring times in corresponding collection (*d*, *P* or *N*), and "*" denotes any word. The meanings of these characters are fixed in the rest of this paper.

The maximum likelihood estimate is an unreasonable one when the amount of training data is small compared to the model size. It is clearly inaccurate to assign zero probability to unseen n-grams. The smoothing describes techniques for adjusting the maximum likelihood estimate to hopefully produce more accurate models.

Dirichlet Prior Smoothing for Unigram

Dirichlet Prior smoothing [10][12] is a linearly interpolated method to the problem of zero probabilities and suitable for unigrams smoothing. Its purpose is to address the estimation bias due to the fact that a document collection is a relatively small amount of data with which to estimate a unigram model. More specifically, it is to discount the *MLE* appropriately and assign non-zero probabilities to n-grams not observed in collection. In terms of unigram model, the smoothing estimation is

$$\Pr_{DP}(w \mid "s") = \begin{cases} \Pr_{\gamma}(w \mid "s") & if\ word\ w\ is\ seen \\ \alpha_s \Pr_{ML}(w \mid C) & otherwise \end{cases} . \tag{6}$$

Where $\Pr_{\gamma}(w \mid "s")$ is the smoothed probability of *w* seen in the collection represented with "*s*". $\Pr_{ML}(w \mid C)$ is the whole corpus (C) language model based on *MLE*, and α_s is a coefficient controlling the probability mass assigned to unseen words, so that all probabilities sum to one. In general, α_s may depend on "*s*". In this study, we exploit the following smoothing formalization,

$$\Pr_{\gamma}(w \mid "s") = \frac{\#(w \; in \; "s") + \mu \Pr_{ML}(w \mid C)}{\#(* \; in \; "s") + \mu}, \tag{7}$$

and

$$\alpha_s = \frac{\mu}{\mu + |C|} . \tag{8}$$

Although Dirichlet Prior smoothing is valid in many NLP tasks, in the sentiment classification of movie review corpus, it only give slight improvement to simple MLE (see the experiment section).

Kenser-Ney Smoothing for Bigram

Kneser and Ney [5] have introduced an extension of absolute discounting where the lower-order distribution that one combines with a higher-order distribution built in a novel manner. To their consideration, a lower-order distribution is a significant factor in the combined model only when few or no counts are present in the higher-order distribution. Following Kneser-Ney smoothing, Stanley F. Chen and Joshua Goodman [10] mathematically motivate Kneser and Ney's algorithm by selecting the lower-order distribution such that the marginal of the higher-order smoothed distribution match the marginal of the training data. Then the Kneser-Ney smoothing performs best compared with other smoothing techniques when given different conditions [10].

To a bigram model, Chen et al select a smoothed distribution Pr_{KN} that satisfies the following constraint on unigram marginals for all w_i:

$$\sum_{w_{i-1}} \Pr_{KN}(w_{i-1}w_i) = \frac{\#(w_i)}{\sum_{w_i} \#(w_i)} . \tag{9}$$

The left-hand side of this equation is the unigram marginal for w_i of the smoothed bigram distribution Pr_{KN} , and the right-hand side is the *MLE* of w_i found in the training data. Therefore, to our smoothing, we assume that the bigram model has the form given in Equation (10),

$$\Pr_{KN}(w_i \mid w_{i-1}, "s") = \frac{\max\{\#(w_{i-1}w_i) - D, 0\}}{\sum_{w_i} \#(w_{i-1}w_i)} + \frac{D}{\sum_{w_i} \#(w_{i-1}w_i)} N_{1+}(w_{i-1}\bullet) \Pr_{KN}(w_i), \quad to\ "s" . \tag{10}$$

In equation (10), D is the fixed discount from observed bigrams and $D = \frac{n_1}{n_1 + 2n_2}$ by Ney's suggestion. Moreover,

$$\Pr_{KN}(w_i) = \frac{N_{1+}(\bullet w_i)}{N_{1+}(\bullet w_{i-1} \bullet)} . \tag{11}$$

For all the meanings of character such as D, N_{1+}, n_1 and n_2, readers can refer to Kneser and Chen's papers[5][10].

3 Document Set and Experiments

Turney [7][8] found movie reviews to be the most difficult of several domains for sentiment classification task, reporting an accuracy of 65.83% on a 120-document set (random-choice baseline: 50%). Herein lies the reason we chose movie reviews for study, and our data source was the Internet Movie Database (IMDB) archive of the *rec.arts.movies.reviews* newsgroup that is adopted in Pang's work [2]. The datasets selected only reviews where the author rating was expressed either with stars or some numerical value. Ratings were extracted and converted into one of two categories: positive (thumb up) and negative (thumb down).

To avoid domination of the corpus by a small number of prolific reviewers, the corpus imposed a limit of fewer than 20 reviews per author per sentiment category, yielding a corpus of 1000 negative and 1000 positive reviews, with a total of more than a hundred reviewers represented. Note that all these original documents were preprocessed by stemming and stop-word removal in our work.

We designed two experiments to investigate SVM and our method. The first was to select the most suitable kernel from linear, polynomial, RBF and sigmoid kernels for sentiment classification. The second was to compare the performance between our method and SVMs. In order to see fair play, all the following experiments selected the features based on word unigrams and bigrams occurring more than 2 times in the 2,000 reviews. The value of a feature is its appearing number.

With respect to the two experiments, we split the 2000 movie reviews into 1200 training samples (600 positive and 600 negative) and 800 test samples (400 positive and 400 negative), and they were both evaluated in average accuracy based on 3-fold cross validation.

SVMs Experiment

We extracted two kinds of input feature sets for SVMs, i.e., unigrams and bigrams. The following experiments compared the performance of SVMs using linear, polynomial, RBF and sigmoid kernels, four conventional learning methods commonly used for text categorization. We used Joachim's SVM^{light} package [11] for training and testing, and other parameters to different kernel functions set to their default values in this package. This experiment aimed at seeing which one is more suitable for the sentiment detection problem. Table 1 outlines the different results of SVMs on the IDMB corpus when different kernel functions are used.

Table 1. Comparison of four kernel functions on the IDMB training and test sets. Linear kernel achieved highest performance on both unigram and bigram features for categorization.

Features	# of features	Linear	Polynomial	Radial Basis Function	Sigmoid
unigrams	13693	**78.21**	59.59	50.09	49.25
bigrams	18602	**73.42**	51.46	51.19	62.00

The best results on the two feature sets come from the SVM using linear kernel. Our language model based method is compared with the SVM using linear kernel.

Language Modeling Approach Experiment

We evaluated the language modeling approach described in section 2 on IMDB collections. As mentioned above, we used unigrams and bigrams models for evaluation. Table 2 is the experiment result.

Table 2. Comparison between language modelling approach and SVMs

Features	# of features	LM-MLE		LM-Smoothing		SVM^{light} (linear kernel)
unigrams	13693	Uni-MLE	82.02	Uni-DP	**84.13**	78.21
bigrams	18602	Bi-MLE	61.62	Bi-KN	**73.80**	73.42

Uni-DP (the smoothed unigram model) performed the best globally on unigrams features set in Table 2, which achieved an average significant improvement of +7.57% compared to the best SVM result. What surprised us is that the simple Uni-MLE could also perform better than SVM-Uni did. On the other hand, the experiment on bigram features showed that the best result of language modeling approach (Bi-KN) was close to the result of SVM and the performance of Bi-MLE was poor.

With respect to the effect of smoothing technique: <1> Dirichlet Prior smoothed unigram model with parameter *u* set to 450 (In our experiment, the best result appeared when we set *u* = 450). Uni-DP performed well for the sentiment classification task but it only slightly improved the performance of Uni-MLE (+2.57%). Although the model based on MLE was inherently poorly estimated, it was not clear that the simple model must be smoothed since the improvement was limited. This phenomenon let us consider that it might be better to find a way of paying more attention to some sensitive concepts to achieve better performance for sentiment classification. <2> Kneser-Ney method smoothed bigram model based on an absolute discount idea, and it did great contribution to estimate a better bigram model leading to a significantly better result than MLE (+19.77%). It obtained the comparable performance to SVM. The reason might be depicted as: in theory, the higher order n-gram model readily approximates the true language model, but for data sparseness, a powerful smoothing mechanism could provide apparent contribution.

4 Conclusion

In this paper, we have presented a new method based on language model for sentiment classification. With respect to this generative sentiment classifier, we represent the "thumb up" and "thumb down" semantic orientation with their corresponding language models estimated from positive and negative collections. When classifying a test document, the distances of its language model from these two sentiment models are compared to determine its sentiment class.

Compared with SVMs, we might conclude as follows in terms of our experimental results: to sentiment classification, when training data is limited, the smoothed low order model, i.e., the unigram model can globally do the best. On the other hand, smoothing technique does great contribution to higher order model that can also achieve comparable performance to SVMs.

The experiments showed the potential power of language modeling approach in this task. We demonstrate that our generative sentiment classifier is applicable by learning the positive and negative semantic orientation efficiently in the supervised manner. This seems to indicate the promising future of language modeling approach for the sentiment detection problem. On the other hand, we stress that the approaches we use are not specific to movie reviews, and it should be easily applicable to other domains when given training data.

The difficulty of sentiment classification is apparent: negative reviews may contain many apparently positive n-grams even while maintaining a strongly negative tone, and the opposite is also common. All classifiers will face this difficulty. To the language modeling approach, our future work will focus on finding a good way to estimate better language models, especially the higher order n-gram models by introducing semantic link between n-grams.

Acknowledgement. This work is supported by NSFC Major Research Program 60496326: Basic Theory and Core Techniques of Non Canonical Knowledge.

References

1. Bo Pang and Lillian Lee: A Sentimental Education: Sentiment Analysis Using Subjectivity Summarization Based on Minimum Cuts. In: Proc. of the 42nd ACL. (2004) 271-278
2. Bo Pang, Lillian Lee and Shivakumar Vaithyanathan: Thumbs up? Sentiment Classification using Machine Learning Techniques. In: Proc. Conf. on EMNLP. (2002)
3. Cover, T. M. and Thomas, J. A.: Elements of Information Theory. Wiley. (1991)
4. Hearst, M.A.: Direction-based text interpretation as an information access refinement. In P. Jacobs (Ed.), Text-Based Intelligent Systems: Current Research and Practice in Information Extraction and Retrieval. Mahwah, NJ: Lawrence Erlbaum Associates. (1992)
5. Kneser,R. & Ney,H: Improved backing-off for m-gram language modeling. In: Proc. of the IEEE Internaltional Conference on Acoustics, Speech and Signal Processing, Detroit, MI,volume 1. May, (1995) 181-184
6. Michael Gamon: Sentiment classification on customer feedback data: noisy data, large feature vectors, and the role of linguistic analysis. In: Proc. the 20th International Conference on Computational Linguistics. (2004)
7. Peter D. Turney: Thumbs up or thumbs down? Semantic orientation applied to unsupervised classification of reviews. In: Proc. of the ACL. (2002)
8. Peter D. Turney and Michael L. Littman: Measuring praise and criticism: Inference of semantic orientation from association. ACM Transactions on Information Systems (TOIS). 21(4), (2003), 315-346
9. Peter D. Turney and Michael L. Littman: Unsupervised learning of semantic orientation from a hundred-billion-word corpus. Technical Report EGB-1094, National Research Council Canada. (2002)
10. S. F. Chen and J. T. Goodman: An Empirical Study of Smoothing Techniques for Language Modeling. Technical Report: TR-10-98, Harvard University. (1998)
11. Thorsten Joachims: Making large-scale SVM learning practical. In Bernhard Scholkopf and Alexander Smola, editors, Advances in Kernel Methods - Support Vector Learning, MIT Press. (1999) 44–56
12. Zhai, C. and Lafferty, J.: A study of smoothing methods for language models applied to ad hoc information retrieval. In Proc. of SIGIR’2001. (2001)