Graph Ranking for Sentiment Transfer

Qiong Wu[1,2], Songbo Tan[1] and Xueqi Cheng[1]
[1]Institute of Computing Technology, Chinese Academy of Sciences, China
[2] Graduate University of Chinese Academy of Sciences, China
{wuqiong,tansongbo}@software.ict.ac.cn, cxq@ict.ac.cn

Abstract

With the aim to deal with sentiment-transfer problem, we proposed a novel approach, which integrates the sentiment orientations of documents into the graph-ranking algorithm. We apply the graph-ranking algorithm using the accurate labels of old-domain documents as well as the "pseudo" labels of new-domain documents. Experimental results show that proposed algorithm could improve the performance of baseline methods dramatically for sentiment transfer.

1 Introduction

With the rapid growth of reviewing pages, sentiment classification is drawing more and more attention (Bai et al., 2005; Pang and Lee, 2008). Generally speaking, sentiment classification can be considered as a special kind of traditional text classification (Tan et al., 2005; Tan, 2006). In most cases, supervised learning methods can perform well (Pang et al., 2002). But when training data and test data are drawn from different domains, supervised learning methods always produce disappointing results. This is so-called cross-domain sentiment classification problem (or sentiment-transfer problem).

Sentiment transfer is a new study field. In recent years, only a few works are conducted on this field. They are generally divided into two categories. The first one needs a small amount of labeled training data for the new domain (Aue and Gamon, 2005). The second one needs no labeled data for the new domain (Blitzer et al., 2007; Tan et al., 2007; Andreevskaia and Bergler, 2008; Tan et al., 2008; Tan et al., 2009). In this paper, we concentrate on the second category which proves to be used more widely.

Graph-ranking algorithm has been successfully used in many fields (Wan et al., 2006; Esuli and Sebastiani, 2007), whose idea is to give a node high score if it is strongly linked with other high-score nodes. In this work, we extend the graph-ranking algorithm for sentiment transfer by integrating the sentiment orientations of the documents, which could be considered as a sentiment-transfer version of the graph-ranking algorithm. In this algorithm, we assign a score for every unlabelled document to denote its extent to "negative" or "positive", then we iteratively calculate the score by making use of the accurate labels of old-domain data as well as the "pseudo" labels of new-domain data, and the final score for sentiment classification is achieved when the algorithm converges, so we can label the new-domain data based on these scores.

2 The Proposed Approach

2.1 Overview

In this paper, we have two document sets: the test data $D^U = \{d_1,...,d_n\}$ where d_i is the term vector of the i^{th} text document and each $d_i \in D^U (i = 1,\ldots,n)$ is unlabeled; the training data $D^L = \{d_{n+1},...d_{n+m}\}$ where d_j represents the term vector of the j^{th} text document and each $d_j \in D^L (j = n+1,\ldots,n+m)$ should have a label from a category set C = {negative, positive}. We assume the training dataset D^L is from the related but different domain with the test dataset D^U. Our objective is to maximize the accuracy of assigning a label in C to $d_i \in D^U$ $(i = 1,\ldots,n)$ utilizing the training data D^L in another domain.

The proposed algorithm is based on the following presumptions:

(1) Let W^L denote the word space of old domain, W^U denote the word space of new domain. $W^L \cap W^U \neq \Phi$.

(2) The labels of documents appear both in the training data and the test data should be the same.

Based on graph-ranking algorithm, it is thought that if a document is strongly linked with positive (negative) documents, it is probably positive (negative). And this is the basic idea of learning from a document's neighbors.

Our algorithm integrates the sentiment orientations of the documents into the graph-ranking algorithm. In our algorithm, we build a graph

Proceedings of the ACL-IJCNLP 2009 Conference Short Papers, pages 317–320,
Suntec, Singapore, 4 August 2009.

whose nodes denote documents and edges denote the content similarities between documents. We initialize every document a score ("1" denotes positive, and "-1" denotes negative) to represent its degree of sentiment orientation, and we call it sentiment score. The proposed algorithm calculates the sentiment score of every unlabelled document by learning from its neighbors in both old domain and new domain, and then iteratively calculates the scores with a unified formula. Finally, the algorithm converges and each document gets its sentiment score. When its sentiment score falls in the range [0, 1] (or [-1, 0]], the document should be classified as "positive (or negative)". The closer its sentiment score is near 1 (or -1), the higher the "positive (or negative)" degree is.

2.2 Score Documents

Score Documents Using Old-domain Information

We build a graph whose nodes denote documents in both D^L and D^U and edges denote the content similarities between documents. If the content similarity between two documents is 0, there is no edge between the two nodes. Otherwise, there is an edge between the two nodes whose weight is the content similarity. The content similarity between two documents is computed with the cosine measure. We use an adjacency matrix U to denote the similarity matrix between D^U and D^L. $U=[U_{ij}]_{nxm}$ is defined as follows:

$$U_{ij} = \frac{d_i \bullet d_j}{\|d_i\| \times \|d_j\|}, \quad i = 1,...,n, j = n+1,...,n+m \qquad (1)$$

The weight associated with term t is computed with $tf_t idf_t$ where tf_t is the frequency of term t in the document and idf_t is the inverse document frequency of term t, i.e. $1+\log(N/n_t)$, where N is the total number of documents and n_t is the number of documents containing term t in a data set.

In consideration of convergence, we normalize U to $\hat{U}$ by making the sum of each row equal to 1:

$$\hat{U}_{ij} = \begin{cases} U_{ij} \Big/ \sum_{j=1}^{m} U_{ij}, & if \sum_{j=1}^{m} U_{ij} \neq 0 \\ 0, & otherwise \end{cases} \qquad (2)$$

In order to find the neighbors (in another word, the nearest documents) of a document, we sort every row of $\hat{U}$ to $\tilde{U}$ in descending order. That is: $\tilde{U}_{ij} \geq \tilde{U}_{ik}$ $(i = 1,\ldots n; j,k = 1,\ldots m; k \geqslant j)$.

Then for $d_i \in D^U$ $(i = 1,\ldots,n)$, $\tilde{U}_{ij}$ $(j = 1,\ldots,K)$ corresponds to K neighbors in D^L. So we can get its K neighbors. We use a matrix $N = [N_{ij}]_{n \times K}$ to denote the neighbors of D^U in old domain, with N_{ij} corresponding to the j^{th} nearest neighbor of d_i.

At last, we can calculate sentiment score s_i $(i = 1,\ldots,n)$ using the scores of the d_i's neighbors as follows:

$$s_i^{(k)} = \sum_{j \in N_{i\bullet}} (\hat{U}_{ij} \times s_j^{(k-1)}), \quad i = 1,...,n \qquad (3)$$

where $i \bullet$ means the i^{th} row of a matrix and $s_i^{(k)}$ denotes the s_i at the k^{th} iteration.

Score Documents Using New-domain Information

Similarly, a graph is built, in which each node corresponds to a document in D^U and the weight of the edge between any different documents is computed by the cosine measure. We use an adjacency matrix $V=[V_{ij}]_{n \times n}$ to describe the similarity matrix. And V is similarly normalized to $\hat{V}$ to make the sum of each row equal to 1. Then we sort every row of $\hat{V}$ to $\tilde{V}$ in descending order, thus we can get K neighbors of $d_i \in D^U$ $(i = 1,\ldots,n)$ from $\tilde{V}_{ij}$ $(j = 1,\ldots K)$, and we use a matrix $M = [M_{ij}]_{n \times K}$ to denote the neighbors of D^U in the new domain. Finally, we can calculate s_i using the sentiment scores of the d_i's neighbors as follows:

$$s_i^{(k)} = \sum_{j \in M_{i\bullet}} (\hat{V}_{ij} \times s_j^{(k-1)}), \quad i = 1,...,n \qquad (4)$$

2.3 Sentiment Transfer Algorithm

Initialization

Firstly, we classify the test data D^U to get their initial labels using a traditional classifier. For simplicity, we use prototype classification algorithm (Tan et al., 2005) in this work.

Then, we give "-1" to $s_i^{(0)}$ if d_i's label is "negative", and "1" if "positive". So we obtain the initial sentiment score vector $S^{(0)}$ for both domain data.

At last, $s_i^{(0)}$ $(i = 1,\ldots,n)$ is normalized as follows to make the sum of positive scores of D^U equal to 1, and the sum of negative scores of D^U equal to -1:

$$s_i^{(0)} = \begin{cases} s_i^{(0)} \Big/ \sum_{j \in D_{neg}^U} (-s_j^{(0)}), & if \quad s_i^{(0)} < 0 \\ s_i^{(0)} \Big/ \sum_{j \in D_{pos}^U} s_j^{(0)}, & if \quad s_i^{(0)} > 0 \end{cases} \quad i = 1,...,n \qquad (5)$$

where D^U_{neg} and D^U_{pos} denote the negative and positive document set of D^U respectively. The same as (5), $s_j^{(0)}$ (j =n+1,…,n+m) is normalized.

Algorithm Introduction

In our algorithm, we label D^U by making use of information of both old domain and new domain. We fuse equations (3) and (4), and get the iterative equation as follows:

$$s_i^{(k)} = \alpha \sum_{j \in N_{i\bullet}} (\hat{U}_{ij} \times s_j^{(k-1)}) + \beta \sum_{h \in M_{i\bullet}} (\hat{V}_{ih} \times s_h^{(k-1)}), i = 1,...,n \quad (6)$$

where $\alpha + \beta = 1$, and α and β show the relative importance of old domain and new domain to the final sentiment scores. In consideration of the convergence, $S^{(k)}$ (S at the k^{th} iteration) is normalized after each iteration.

Here is the complete algorithm:

1. Classify D^U with a traditional classifier. Initialize the sentiment score s_i of $d_i \in D^U \cup D^L$ (i = 1,…n+m) and normalize it.
2. Iteratively calculate the $S^{(k)}$ of D^U and normalize it until it achieves the convergence:

$$s_i^{(k)} = \alpha \sum_{j \in N_{i\bullet}} (\hat{U}_{ij} \times s_j^{(k-1)}) + \beta \sum_{h \in M_{i\bullet}} (\hat{V}_{ih} \times s_h^{(k-1)}), i = 1,...,n$$

$$s_i^{(k)} = \begin{cases} s_i^{(k)} \Big/ \sum_{j \in D^U_{neg}} (-s_j^{(k)}), & if \quad s_i^{(k)} < 0 \\ s_i^{(k)} \Big/ \sum_{j \in D^U_{pos}} s_j^{(k)}, & if \quad s_i^{(k)} > 0 \end{cases} \quad i = 1,...,n$$

3. According to $s_i \in S$ (i = 1,…,n), assign each $d_i \in D^U$ (i = 1,…n) a label. If s_i is between -1 and 0, assign d_i the label "negative"; if s_i is between 0 and 1, assign d_i the label "positive".

3 EXPERIMENTS

3.1 Data Preparation

We prepare three Chinese domain-specific data sets from on-line reviews, which are: Electronics Reviews (Elec, from http://detail.zol.com.cn/), Stock Reviews (Stock, from http://blog.sohu.com /stock/) and Hotel Reviews (Hotel, from http://www.ctrip.com/). And then we manually label the reviews as "negative" or "positive".

The detailed composition of the data sets are shown in Table 1, which shows the name of the data set (DataSet), the number of negative reviews (Neg), the number of positive reviews (Pos), the average length of reviews (Length), the number of different words (Vocabulary) in this data set.

DataSet	Neg	Pos	Length	Vocabulary
Elec	554	1,054	121	6,200
Stock	683	364	460	13,012
Hotel	2,000	2,000	181	11,336

Table 1. Data sets composition

We make some preprocessing on the datasets. First, we use ICTCLAS (http://ictclas.org/), a Chinese text POS tool, to segment these Chinese reviews. Second, the documents are represented by vector space model.

3.2 Evaluation Setup

In our experiment, we use prototype classification algorithm (Tan et al., 2005) and Support Vector Machine experimenting on the three data sets as our baselines separately. The Support Vector Machine is a state-of-the-art supervised learning algorithm. In our experiment, we use LibSVM (www.csie.ntu.edu.tw/~cjlin/libsvm/) with a linear kernel and set all options by default.

We also compare our algorithm to Structural Correspondence Learning (SCL) (Blitzer et al., 2007). SCL is a state-of-the-art sentiment-transfer algorithm which automatically induces correspondences among features from different domains. It identifies correspondences among features from different domains by modeling their correlations with pivot features, which are features that behave in the same way for discriminative learning in both domains. In our experiment, we use 100 pivot features.

3.3 Overall Performance

In this section, we conduct two groups of experiments where we separately initialize the sentiment scores in our algorithm by prototype classifier and Support Vector Machine.

There are two parameters in our algorithm, K and α (β can be calculated by 1-α). We set the parameters K and α with 150 and 0.7 respectively, which indicates we use 150 neighbors and the contribution from old domain is a little more important than that from new domain. It is thought that the algorithm achieves the convergence when the changing between the sentiment score s_i computed at two successive iterations for any $d_i \in D^U$ (i = 1,…n) falls below a given threshold, and we set the threshold 0.00001 in this work.

Table 2 shows the accuracy of Prototype, LibSVM, SCL and our algorithm when training data and test data belong to different domains.

Our algorithm is separately initialized by Prototype and LibSVM.

	Baseline		SCL	Proposed Algorithm	
	Prototype	LibSVM		Prototype+ OurApproach	LibSVM+ OurApproach
Elec->Stock	0.6652	0.6478	**0.7507**	0.7326	0.7304
Elec->Hotel	0.7304	0.7522	**0.7750**	0.7543	0.7543
Stock->Hotel	0.6848	0.6957	**0.7683**	0.7435	0.7457
Stock->Elec	0.7043	0.6696	0.8340	**0.8457**	0.8435
Hotel->Stock	0.6196	0.5978	0.6571	**0.7848**	**0.7848**
Hotel->Elec	0.6674	0.6413	0.7270	**0.8609**	**0.8609**
Average	0.6786	0.6674	0.7520	**0.7870**	0.7866

Table 2. Accuracy comparison of different methods

As we can observe from Table 2, our algorithm can dramatically increase the accuracy of sentiment-transfer. Seen from the 2nd column and the 5th column, every accuracy of the proposed algorithm is increased comparing to Prototype. The average increase of accuracy over all the 6 problems is 10.8%. Similarly, the accuracy of our algorithm is higher than LibSVM in every problem and the average increase of accuracy is 11.9%. The great improvement comparing with the baselines indicates that the proposed algorithm performs very effectively and robustly.

Seen from Table 2, our result about SCL is in accord with that in (Blitzer et al., 2007) on the whole. The average accuracy of SCL is higher than both baselines, which convinces that SCL is effective for sentiment-transfer. However, our approach outperforms SCL: the average accuracy of our algorithm is about 3.5 % higher than SCL. This is caused by two reasons. First, SCL is essentially based on co-occurrence of words (the window size is the whole document), so it is easily affected by low frequency words and the size of data set. Second, the pivot features of SCL are totally dependent on experts in the field, so the quality of pivot features will seriously affect the performance of SCL. This improvement convinces us of the effectiveness of our algorithm.

4 Conclusion and Future Work

In this paper, we propose a novel sentiment-transfer algorithm. It integrates the sentiment orientations of the documents into the graph-ranking based method for sentiment-transfer problem. The algorithm assigns a score for every document being predicted, and it iteratively calculates the score making use of the accurate labels of old-domain data, as well as the "pseudo" labels of new-domain data, finally it labels the new-domain data as "negative" or "positive" basing on this score. The experiment results show that the proposed approach can dramatically improve the accuracy when transferred to a new domain.

In this study, we find the neighbors of a given document using cosine similarity. This is too general, and perhaps not so proper for sentiment classification. In the next step, we will try other methods to calculate the similarity. Also, our approach can be applied to multi-task learning.

5 Acknowledgments

This work was mainly supported by two funds, i.e., 0704021000 and 60803085, and one another project, i.e., 2004CB318109.

References

B. Pang and L. Lee. 2008. Opinion mining and sentiment analysis. Foundations and Trends in Information Retrieval, 2008

S. Tan, X. Cheng, M. Ghanem, B. Wang and H. Xu. 2005. A Novel Refinement Approach for Text Categorization. In *Proceedings of CIKM 2005*.

S. Tan. 2006. An Effective Refinement Strategy for KNN Text Classifier. Expert Systems With Applications. Elsevier. 30(2): 290-298.

B. Pang, L. Lee, and S. Vaithyanathan. 2002. Thumbs up? Sentiment classification using machine learning techniques. In *Proceedings of EMNLP, 2002*.

X. Bai, R. Padman and E. Airoldi. 2005. On learning parsimonious models for extracting consumer opinions. In *Proceedings of HICSS 2005*.

A. Aue and M. Gamon. 2005. Customizing sentiment classifiers to new domains: a case study. In *Proceedings of RANLP 2005*.

J. Blitzer, M. Dredze, and F. Pereira. 2007. Biographies, Bollywood, Boom-boxes and Blenders: Domain adaptation for sentiment classification. In *Proceedings of ACL 2007*.

S. Tan, G. Wu, H. Tang and X. Cheng. 2007. A novel scheme for domain-transfer problem in the context of sentiment analysis. In *Proceedings of CIKM 2007*.

S. Tan, Y. Wang, G. Wu and X. Cheng. 2008. Using unlabeled data to handle domain-transfer problem of semantic detection. In *Proceedings of SAC 2008*.

S. Tan, X. Cheng, Y. Wang, H. Xu. 2009. Adapting Naive Bayes to Domain Adaptation for Sentiment Analysis. In *Proceedings of ECIR 2009*.

A. Esuli, F. Sebastiani. 2007. Random-walk models of term semantics: An application to opinion-related properties. In *Proceedings of LTC 2007*.

X. Wan, J. Yang and J. Xiao. 2006. Using Cross-Document Random Walks for Topic-Focused Multi-Document Summarization. In *Proceedings of WI 2006*.

A. Andreevskaia and S. Bergler. 2008. When Specialists and Generalists Work Together: Overcoming Domain Dependence in Sentiment Tagging. In *Proceedings of ACL 2008*.