A Dictionary-Based Approach to Identifying Aspects Implied by Adjectives for Opinion Mining

Geli Fei [1] *Bing Liu* [1] *Meichun Hsu* [2] *Malu Castellanos* [2] *Riddhiman Ghosh* [2]
(1) Department of Computer Science, University of Illinois at Chicago, Chicago, USA
(2) HP Labs, Palo Alto, California, USA
gfei2@uic.edu, liub@cs.uic.edu
{meichun.hsu, malu.castellanos, riddhiman.ghosh}@hp.com

ABSTRACT

One of the central problems of opinion mining is to extract aspects of entities or topics that have been evaluated in an opinion sentence or document. Much of the existing research focused on extracting explicit aspects which are nouns and nouns phrases that have appeared in sentences, e.g., *price* in "*The price of this bike is very high*." However, in many cases, people do not explicitly mention an aspect in a sentence, but the aspect is implied, e.g., "*This bike is expensive*," where *expensive* indicates the *price* aspect of the bike. Although there are some existing works dealing with the problem, they all used the corpus-based approach, which has several shortcomings. In this paper, we propose a dictionary-based approach to address these shortcomings. We formulate the problem as collective classification. Experimental results show that the proposed approach is effective and produces significantly better results than strong baselines based on traditional supervised classification.

KEYWORDS: Implied Aspects or Topics, Opinion Mining, Sentiment Analysis

1 Introduction

In sentiment analysis, the task of aspect extraction is to identify aspects of entities or topics on which opinions have been expressed (Hu and Liu 2004). For example, in the sentence "*The picture quality of this camera is great*," *picture quality* is an aspect of the camera. In most cases, aspects appear explicitly in sentences, e.g., *picture quality*. Such aspects are *called explicit aspects* (Hu and Liu 2004). However, in many other cases, they do not appear, but are implied. For instance, the sentence "*This is an expensive bike*" gives a negative opinion about the *price* aspect. However, *price* is not in the sentence, but it is clearly implied. *Price* is called an *implicit aspect* (Liu, 2010). Price is also called an attribute of *expensive* in lexical semantics (Almuhareb, 2006). In this paper, we will use the terms *aspect* and *attribute* interchangeably as they mean the same thing in our context. Since aspects or attributes used in this work are nouns, we also call them *aspects/attribute nouns*.

Implicit aspects can be indicated by many types of expressions, e.g., adjectives, adverbs, verbs and their phrases. This paper focuses on opinion adjectives. Although there are general opinion adjectives which can describe anything, e.g., *good* and *bad*, most adjectives describe some specific attributes of entities. The goal of this work is to identify attribute nouns of each adjective, e.g., to identify *price*, *cost*, etc., for adjective *expensive*.

There are some existing works that tried to find implicit aspects indicated by adjectives (Su et al., 2008; Hai et al., 2011). They all depend on co-occurrences of adjectives and explicit attribute nouns in sentences in a corpus. There are also some relevant works in lexical semantics, which also use corpus-based techniques (Almuhareb and Poesio 2004; Hartung and Frank, 2010; Hartung and Frank, 2011). The corpus-based approach is useful for finding context specific mappings of adjectives and attributes because an adjective can have multiple senses. In a specific domain or context, it takes only a specific sense (which needs to be discovered). However, the corpus-based approach alone also has some weaknesses:

1. It is hard to discover attributes that do not co-occur with their adjectives. For example, in English, people don't say "*The price of iPhone is expensive*." Instead, they say "*iPhone is expensive.*" It is thus hard for a corpus-based approach to find *price* for *expensive*.
2. Even if an adjective and one of its attribute nouns do appear in a corpus, due to the limited corpus size, they may not co-occur in many sentences to be associated reliably.
3. If one wants to find all attribute nouns for each adjective, it is also difficult due to the corpus size limit because not all adjectives or all attributes may appear in a corpus.

In this work, we propose a dictionary-based approach which complements the corpus-based approach and can address these problems. The first and the second problems are tackled because dictionaries typically define adjectives using their attributes. For example, *expensive* is defined as "*Marked by high prices*" in thefreedictionary.com. The third problem is also addressed because dictionaries are not restricted by any specific corpus. We can work on every adjective in a dictionary. Since not all attribute nouns of an adjective may appear in a dictionary, we use multiple dictionaries for better coverage. To our knowledge, this is the first dictionary-based approach. It finds all attribute nouns for an adjective.

We propose to solve the problem using a relational learning method called *collective classification* (Sen et al. 2008), which can take advantage of rich lexical relationships of words (e.g., synonyms, antonyms, hyponym and hypernym) for classification. Our evaluation shows that collective classification outperforms traditional classification significantly.

2 The Proposed Approach

Our proposed method consists of three steps:

1. Given a set of adjectives $A = \{A_1, A_2, \ldots, A_r\}$, crawl the online dictionaries for their glosses.
2. For each adjective $A_i \in A$, perform POS tagging of its glosses and extract nouns from them. These nouns are regarded as the candidate attribute nouns C_i for adjective A_i.
3. Classify each candidate attribute noun $c_{ij} \in C_i$ to one of the two classes, *attribute noun* or *not attribute noun*, of A_i. This step uses a collective classification algorithm to exploit the lexical relationships of words in dictionaries to build more accurate classifiers.

Since the first two steps are straightforward, the rest of the paper focuses on step 3.

2.1 Problem Formulation and Solution

In traditional supervised learning, each instance is drawn independently of others (Mitchell, 1997). However, in many real-life data, instances are not independent of each other. Such data is often represented as a graph where nodes are instances and links are their relations. The classification of one node can influence its neighboring nodes. This type of classification is called *collective classification* (Sen et al., 2008) as opposed to the instance-based classification. We formulate the proposed problem as collective classification.

Each instance in our data denotes a pair with an adjective A_i and one of its candidate attribute nouns c_{ij}, i.e., (A_i, c_{ij}). Due to the relational features (which will be detailed later), we use a graph representation of instances, with a set of nodes (pairs), $V = \{(A_i, c_{ij}) \mid c_{ij} \in C_i, A_i \in A\}$, and a neighborhood function N, where $N_{ij} \subseteq V - \{(A_i, c_{ij})\}$. Each node (a pair (A_i, c_{ij})) in V is represented with a vector $\mathbf{x}_{ij}$ of features, $f_1, f_2, \ldots, f_n$, and is associated with a class label y_{ij} in the domain of {positive, negative}. The positive class means *attribute noun*, and the negative class means *not attribute noun*. V is further divided into two sets of nodes: L, labeled nodes, and U, unlabeled nodes. Our task is to predict the label for each node $u_{ij} \in U$.

A collective classification algorithm called the *iterative classification algorithm* (ICA) (Sen et al. 2008) is employed to solve this problem. ICA is given in Figure 1. Its training process (not in Figure 1) trains a classifier h just like traditional supervised learning, using the labeled set L with all features. The classification (or testing) step is the core of this algorithm.

In testing, the learned classifier h assigns a class label to each node $u_{ij} \in U$ in the test data (lines 1-4). Line 2 computes the feature vector $\mathbf{x}_{ij}$ for u_{ij}. This (and also line 8) is an important step of this algorithm which makes it different from the classic supervised learning. It computes all the relational features for u_{ij} using the neighbors of u_{ij}. However, line 2 is slightly different from line 8 as in line 2 not all nodes have been assigned class labels, so we compute $\mathbf{x}_{ij}$ based on the intersection of the labeled nodes (L) and u_{ij}'s neighbors. Line 3 uses h to assign a class (y_{ij}) to node u_{ij}. Lines 1-4 are considered as the initialization step.

After initialization, the classifier is run iteratively (lines 5-11) until the class labels of all nodes no longer change. The iterations are needed because some relational features of a node depend on the class labels of its neighbors. Such labels are assigned in each iteration and may change from one iteration to the next. In each iteration (lines 6-10), the algorithm first generates an ordering of nodes to be classified. We order them randomly in order to reduce bias as the random ordering makes the process stochastic. Line 8 does the same job as line 2. Line 9 does the same job as line 3. Classifier h does not change in the iterations.

Algorithm ICA - Iterative classification

```
1.  for each node $u_{ij} \in U$      // each node is a pair2.
    compute $\mathbf{x}_{ij}$ using only $L \cap N_{ij}$
3.     $y_{ij} \leftarrow h(\mathbf{x}_{ij})$
4.  endfor
5.  repeat // iterative classification
6.     generate an ordering $O$ over pairs in $U$
7.      for each node $o_{ij} \in O$ do8.         compute $\mathbf{x}_{ij}$ using
    current assignments to $N_{ij}$
9.         $y_{ij} \leftarrow h\ (\mathbf{x}_{ij})$
10.    endfor
11. until all class labels do not change
```

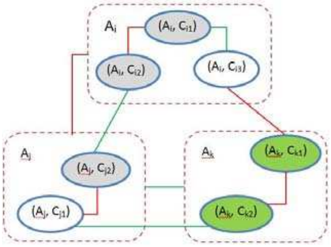

Figure 2. An example of a graph of word relations and an ICA iteration

Figure 2 shows a simplified example of a graph based on some relationships of words. It is also a snapshot of an iteration of ICA. Each oval node denotes an instance (an adjective and attribute pair). A dashed box encloses the pairs that belong to the same adjective. A link between two oval nodes denotes a relationship between two (candidate) attribute nouns, and a link between two dashed boxes denotes a relationship between two adjectives. Green lines denote synonym and red lines denote antonym. The green shaded nodes denote those labeled pairs, the grey shaded nodes denote those candidate attribute nouns whose labels have been predicted (unlabeled at the beginning), whereas un-shaded oval nodes denote those candidate attribute nouns whose labels are yet to be predicted in the iteration. In the figure, adjectives A_k and A_j are synonyms, attribute noun c_{k2} (labeled) and candidate attribute noun c_{j1} are synonyms, and candidate attribute nouns c_{j1} and c_{j2} are antonyms. In the previous iteration, ICA has predicted/labeled c_{j2} as an attribute noun of A_j. Since c_{j2}, c_{j1} and c_{k2} are related, the label of c_{j1} will be affected by the labels of c_{j2} and c_{k2} in this iteration.

2.2 Useful Relations

In this work, we consider two kinds of relations for adjectives: synonym and antonym, and four kinds of relations for nouns: synonym, antonym, hypernym and hyponym. Using them, we created two sets of relational features, *static* (relational) *features* and *dynamic* (relational) *features*. Static features are not affected by the classification process in testing. Dynamic features are affected by the classification process, i.e., the values of these features can change during the testing phase because they depend on the predicted labels of its neighbours (which are also candidate attribute noun and adjective pairs) (see Section 2.5). Finally, we have three sets of features: (1) local features (these are the traditional features about each instance itself), (2) static relational features, and (3) dynamic relational features.

2.3 Local Features

The local features (L1, ..., L6) are only about the adjective-noun pair (A_i, c_{ij}) itself:

L1. Word *n*-grams: These are traditional n-grams of words in the glosses of each adjective A_i.

L2. Part of speech (POS) *n*-grams: n-grams of POS tags. These are also traditional features.

L3. Number of times that candidate attribute noun c_{ij} appears in the glosses for adjective A_i in all dictionaries. Intuitively, the more times it appears, the more likely it is a true attribute.

L4. Diversity of candidate nouns in C_i for adjective A_i: The idea is that if the candidate words are too numerous and all different, then they are less likely to be true attribute nouns. Entropy is one of the methods for measuring diversity. Let n_{ij} be the frequency that the candidate attribute noun $c_{ij} \in C_i$, as well as c_{ij}'s synonyms and antonyms, occur in the glosses of A_i in all dictionaries. We call a set of words formed by c_{ij} and its synonyms and antonyms in C_i a *semantic group* for c_{ij}. Let m be the number of semantic groups formed by the words in C_i. Let T_i be the occurrence count of A_i's candidate attribute nouns in all dictionaries. Let p_{ij} (= n_{ij} /T_i) be the probability of occurrence of the candidate nouns in c_{ij}'s semantic group in the dictionaries. The diversity (entropy) of C_i is defined as:

$$diversity\ (C_i) = -\sum_{j=1}^{m} p_{ij} \log\ p_{ij} \tag{1}$$

L5. Similarity of candidate attribute noun c_{ij} and its adjective A_i. This is the number of same letters (m_{ij}) in their prefixes normalized by the maximum length (len(.)) of the two words,

$$sim\ (c_{ij}, A_i) = \frac{m_{ij}}{\max(\ len\ (c_{ij}), len\ (A_i))} \tag{2}$$

We use this feature because in some cases a noun is turned into an adjective with ending changes, e.g., *style* (c_{ij}) and *stylistic* (A_i) (their similarity is 4/9).

L6. Frequent POS sequence patterns mined from the POS tags of q (= 5) words right before each candidate attribute noun c_{ij} in a gloss, using a sequence pattern mining algorithm (Srikant and Agrawal, 1996). All the discovered patterns are used as features. Note that POS patterns are not POS n-grams because a pattern can skip POS tags but a POS n-gram is a sequence of consecutive POS tags. For pattern discovery, every gloss sentence containing c_{ij} generate a POS tag sequence for mining. For testing, when multiple glosses containing c_{ij} (we use multiple dictionaries), as long as the POS tags of the q word before one occurrence of c_{ij} satisfies the pattern, the feature for the pattern is set to 1; otherwise 0.

2.4 Static Relational Features

To define relational features, we first need to define some relations. Let R_s be a binary synonym function and R_a be a binary antonym function on the set of all adjectives or candidate attribute nouns. For w_i, $w_j \in A$ (all adjectives) or w_i, $w_j \in C$ (all candidate attribute nouns), if $R_s(w_i, w_j) = 1$, w_i and w_j are synonyms. If $R_s(w_i, w_j) = 0$, w_i and w_j are not synonyms. If $R_a(w_i, w_j) = 1$, w_i and w_j are antonyms. If $R_a(w_i, w_j) = 0$, w_i and w_j are not antonyms. Similarly, we have R_{hyper} (hypernym) and R_{hypo} (hyponym) on the set of all candidate attribute nouns C. We also assume that both R_s and R_a are symmetric, which means that for all w_i, $w_j \in A$ or w_i, $w_j \in C$, $R_s(w_i, w_j)$ implies $R_s(w_j, w_i)$, and $R_a(w_i, w_j)$ implies $R_a(w_j, w_i)$.

We now present the static relational features. Let g_{id} be the glosses in the *d-th* dictionary for adjective A_i. Let $E(c_{ij}, g_{id})$ be a function that returns the number of times that c_{ij} occurs in g_{id}. For each node (or pair) (A_i, c_{ij}), we have the following 7 static relational features:

S1-S4. These four features represent respectively the number of times that c_{ij}'s synonyms, antonyms, hypernyms and hyponyms appear in the glosses of A_i in the dictionaries,

$$\sum_{k=1}^{|S|} \sum_{d=1}^{H} R(c_{ik}, c_{ij}) E(c_{ik}, g_{id}) \tag{3}$$

where S is the set of synonyms, antonyms, hypernyms or hyponyms of $c_{ij} \in C_i$ and H is the number of dictionaries, $R \in \{R_s, R_a, R_{hyper}, R_{hypo}\}$. These relationships are extracted from the WordNet. These features are relational because they are related to other nodes in the graph as each synonym, antonym, hypernym or hyponym of c_{ij} in S that appears in the glosses of a dictionary also generates an instance (or a node) in the data. And the reason we call these relational features static is because they don't change during the testing phase. These features are used because the more times that c_{ij}'s synonyms, antonyms, hypernyms or hyponyms appear in the glosses of adjective A_i, the more likely c_{ij} is a true attribute noun of A_i.

S5-S6. These two features represent respectively the total number of times that c_{ij} appears in the glosses of A_i's synonyms and antonyms,

$$\sum_{k=1}^{|S|}\sum_{d=1}^{H} R(A_i, A_k)E(c_{ij}, g_{kd}) \tag{4}$$

where S is the set of synonyms or antonyms of A_i in set A, and $R \in \{R_s, R_a\}$.

S7. The number of times that c_{ij} appears in the glosses of other adjectives which are neither synonym nor antonym of A_i. This feature can be calculated as follows,

$$\sum_{k=1}^{|S|}\sum_{d=1}^{H}(1-R_s(A_i, A_k))(1-R_a(A_i, A_k))E(c_{ij}, g_{kd}) \tag{5}$$

where S is the set of all adjectives in the data. The intuition is that the more c_{ij} appears, the less likely it is a real attribute for A_i.

2.5 Dynamic Relational Features

Dynamic relational features mean that their values can change during the testing phase because they are calculated based on the classified labels of neighboring nodes. That is, these features let the system see how the neighboring nodes of the current node are classified, which affects the classification of the current node.

For each $c_{ij} \in C_i$ of adjective A_i, $Y(c_{ij}, A_i)$ denotes the class label of node (A_i, c_{ij}). If the node is classified as positive (we also say that c_{ij} is classified as an attribute noun of adjective A_i), $Y(c_{ij}, A_i) = 1$; otherwise $Y(c_{ij}, A_i) = 0$. We have the following 14 dynamic relational features:

D1-D4. These four features represent respectively the number of times that c_{ij}'s synonyms, antonyms, hypernyms and hyponyms are classified as attribute nouns for A_i,

$$\sum_{k=1}^{|S|} R(c_{ik}, c_{ij})Y(c_{ik}, A_i) \tag{6}$$

where S is the set of synonyms, antonyms, hypernyms or hyponyms of $c_{ij} \in C_i$, and $R \in \{R_s, R_a, R_{hyper}, R_{hypo}\}$. We use these features because if a synonym, antonym, hypernym or hyponym of c_{ij} is an attribute noun for A_i then c_{ij} is also likely to be such a noun for A_i.

D5-D6. These two features represent respectively the number of times that c_{ij} is classified as attributes for A_i's synonyms and antonyms,

$$\sum_{k=1}^{|S|} R(A_k, A_i)Y(c_{ij}, A_k) \tag{7}$$

where S is the set of synonyms or antonyms of $A_i \in A$ and $R \in \{R_s, R_a\}$.

D7-D14. These eight features represent respectively the number of times that c_{ij}'s synonyms, antonyms, hypernyms and hyponyms are classified as attribute nouns for A_i's synonyms and antonyms,

$$\sum_{p=1}^{|T|}\sum_{q=1}^{|S|} R_C(c_{ip}, c_{ij})R_A(A_q, A_i)Y(c_{ip}, A_q) \tag{8}$$

where S is the set of synonyms or antonyms of A_i, T is the set of synonyms, antonyms, hypernyms or hyponyms of $c_{ij} \in C_i$, and $R_C \in \{R_s, R_a, R_{hyper}, R_{hypo}\}$, $R_A \in \{R_s, R_a\}$. So we obtain a total of 8 dynamic features.

local features	**Accuracy**	**F-score**
Best local feature combination (L3, L4, L5, L6)	0.689	0.701

Table 1 – Usefulness of different local features

	Feature sets	**Logistic Regression**				**SVM**			
		Prec	**Rec**	**F-score**	**Acc**	**Prec**	**Rec**	**F-score**	**Acc**
Strategy 1	Local features (traditional learning)	0.689	0.723	0.701	0.689	0.731	0.616	0.654	0.695
	Local+static features	0.715	0.722	0.716	0.710	0.750	0.627	0.675	0.708
	Local+dynamic features	0.725	**0.783**	**0.746**	0.730	0.741	0.668	0.700	0.721
	All features	**0.747**	0.742	0.743	0.732	0.756	0.621	0.676	0.710
Strategy 2	ICA (all features)	**0.791**	0.675	0.725	0.736	0.823	0.518	0.624	0.700
	ICA (local+dynamic features)	0.750	**0.766**	**0.754**	**0.742**	0.792	0.594	0.670	0.717

Table 2 – Average Precision, Recall, F-score and Accuracy results over 10-fold cross-validations

3 Experimental Results

We now evaluate the proposed technique. First, we compare the results of different feature sets, i.e., local features, static relational features, and dynamic relational features, and also two learning strategies. Note that using only local features is the traditional supervised classification. Second, we compare our results with WordNet in terms of attribute coverage.

3.1 Experiment Settings

Datasets: Our data were extracted from 5 online dictionaries: *Dictionary.com*, *The Free Dictionary*, *Longman Dictionary of Contemporary English*, *Your Dictionary*, and *The Free Merriam-Webster Dictionary*. For opinion adjectives, we used a subset of 310 adjectives from the opinion lexicon of Hu and Liu (2004)[1]. From each dictionary, we extracted the glosses of these adjectives. The Stanford POS Tagger[2] (Toutanova et al., 2003) was used to find nouns. The nouns from each adjective's gloss were considered as its candidate attribute nouns.

Altogether 4410 adjective-noun pairs from 310 adjectives were annotated by two human labelers. Kappa (κ) gave κ = 0.77 (substantial agreement (Landis and Koch, 1977)). As a 2-class classification problem, we treat *attribute noun* as the positive class, and *not attribute noun* as the negative class. The distribution of the positive and negative classes is 48% and 52% respectively. All classification results were obtained through 10-fold cross-validations.

3.2 Results and Discussions

We first assess the usefulness of different local features. Traditional classification is applied to these features. Table 1 gives the best local feature combination (L3, L4, L5, and L6). Word n-grams and POS n-grams were found not so useful. POS n-grams also perform worse than POS patterns (due to space limitations, we cannot show the detailed results) because n-grams are consecutive POS tags, while POS patterns do not have to be consecutive. This makes POS patterns better able to capture the regularities in the text. Next we evaluate the collective classification based on the best set of local features and all static and dynamic features. Two classification strategies were examined.

[1] http://www.cs.uic.edu/~liub /FBS/sentiment-analysis.html
[2] http://nlp.stanford.edu/software/tagger.shtml

Strategy 1 (two stages): The first stage simply builds a local classifier using the local features or a combination of local and static relational features to classify each node. The results serve as the initialization for stage two. In the second stage, dynamic relational features are added to run the ICA algorithm in Figure 1 without the first 4 lines.

Strategy 2 (one stage): We simply train with both local and relational features. The trained classifier is then applied to classify the test data using the ICA algorithm in Figure 1.

Table 2 shows the results of for each strategy and each feature set for Logistic Regression (LR) and SVM. For LR, we used the Lingpipe system (http://alias-i.com/lingpipe/). For SVM, we used *SVM^{light}* (http://svmlight. joachims.org/). From the table, we can see that LR performs better than SVM in general. Our discussions and comparisons below are thus based on LR. Table 2 also allows us to make the following observations:

1. "Local" features perform the worst (traditional classification). With the addition of either the two sets of relational features, the results improve. The dynamic relational features are most useful. We can say that the results of collective classification are superior.
2. For strategy 1, we see that "local+static" outperforms "local" features. Using all features is even better. "local+dynamic" features gives us the best F-score.
3. For strategy 2, using all features again performs better than only "local" features. Using "local+dynamic" gives both the best F-score and accuracy among all experiments.

Compare with WordNet: We now compare our method with WordNet, which can retrieve attributes given an adjective. Table 3 shows the comparison results. Column 2 gives the average number of correct attributes found by our system over 3-fold cross validation and by WordNet respectively. Our method can find far more attribute nouns than WordNet. Although WordNet has 100% precision (as it was manually compiled), the recall is so low. Many adjectives have no attribute nouns in WordNet, e.g., it gives no attribute for *expensive*.

	No. of correct attributes found	**Prec.**	**Rec.**	**F-score**
WordNet	53	100%	7.9%	0.146
Our method	522	76.3%	77.3%	0.768

Table 3 – Comparison results of WordNet and our method (3-fold cross-validation)

5 Conclusion

This paper studied the problem of mining attribute nouns of opinion adjectives. A dictionary-based approach was proposed. To our knowledge, this is the first work using such an approach. Existing works are all based on corpuses. To solve the problem, we formulated it as collective classification as words are related through many lexical relations. Such relations can be exploited to produce better classifiers. Our evaluation showed that collective classification using dynamic relational features performed significantly better than traditional classification. It also performs dramatically better than WordNet. Finally, we note that there are two related approaches used in finding opinion words: the corpus-based approach (e.g., Hazivassiloglou and McKeown, 1997; Wilson et al., 2005; Kanayama and Nasukawa, 2006; Ding et al., 2008; Choi and Cardie, 2008; Wu and Wen, 2010) and the dictionary based approach (e.g., Hu and Liu 2004; Kim and Hovy, 2004; Kamps et al., 2004; Esuli and Sebastiani, 2005; Andreevskaia and Bergler, 2006; Blair-Goldensohn et al., 2008; Hassan and Radev, 2010). Although the two approaches are analogous to the two corresponding approaches for the attribute discovery of adjectives, the two tasks are entirely different.

References

Almuhareb, A. 2006. *Attributes in Lexical Acquisition*. Ph.D. *Dissertation*, Department of Computer Science, University of Essex.

Almuhareb, A and M. Poesio. 2004. Attribute-Based and Value-Based Clustering: An Evaluation. *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Andreevskaia, A. and S. Bergler. 2006. Mining WordNet for fuzzy sentiment: Sentiment tag extraction from WordNet glosses. *Proceedings of Conference of the European Chapter of the Association for Computational Linguistic*s (EACL-06). 2006.

Blair-Goldensohn, S., K. Hannan, and R. McDonald, Tyler Neylon, George A. Reis, and Jeff Reynar. 2008. Building a sentiment summarizer for local service reviews. *Proceedings of WWW-2008 Workshop on NLP in the Information Explosion Era*.

Choi, Y. and C. Cardie. Learning with compositional semantics as structural inference for subsentential sentiment analysis. *Proceedings of Conference on Empirical Methods in Natural Language Processing* (EMNLP-2008). 2008.

Ding, X., B. Liu, and P. S. Yu. 2008. A holistic lexicon-based approach to opinion mining. *Proceedings of the Conference on Web Search and Web Data Mining* (WSDM-2008).

Esuli, A. and F. Sebastiani. Determining the semantic orientation of terms through gloss classification. *Proceedings of ACM International Conference on Information and Knowledge Management* (CIKM-2005). 2005.

Hai, Z., K. Chang, and J. Kim. 2011. Implicit feature identification via co-occurrence association rule mining. *Computational Linguistics and Intelligent Text Processing*, 2011: p. 393-404.

Hartung, M and A. Frank, 2010. A Structured Vector Space Model for Hidden Attribute Meaning in Adjective-Noun Phrases. *Coling* 2010.

Hartung, M and A. Frank, 2011. Exploring Supervised LDA Models for Assigning Attributes to Adjective-Noun Phrases. *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.

Hassan, A. and D. Radev. 2010. Identifying text polarity using random walks. *Proceedings of Annual Meeting of the Association for Computational Linguistics* (ACL-2010).

Hatzivassiloglou, V. and K. R. McKeown. 1997. Predicting the semantic orientation of adjectives. *Proceedings of Annual Meeting of the Association for Computational Linguistics* (ACL-1997).

Hu, M. and Liu, B. 2004. Mining and summarizing customer reviews. *Proceedings of SIGKDD International Conference and Knowledge Discovery and Data Mining.*

Kamps, J., M. Marx, R. J. Mokken, and M. De Rijke. 2004. Using WordNet to measure semantic orientation of adjectives. Proc. of LREC-2004.

Kanayama, H. and T. Nasukawa. 2006. Fully automatic lexicon expansion for domain-oriented sentiment analysis. *Proceedings of Conference on Empirical Methods in Natural Language Processing* (EMNLP-2006).

Kim, S-M and E. Hovy. 2004. Determining the sentiment of opinions. *Proceedings of International Conference on Computational Linguistics* (COLING-2004).

Landis, J. R. and G. G. Koch. 1977. The measurement of observer agreement for categorical data. *Biometrics*.

Liu, B. 2010. Sentiment analysis and subjectivity. In Handbook of Natural Language Processing, edited by Indurkhya, N and Damerau, F. J.

Mitchell, T. 1997. *Machine Learning*, McGraw Hill.

Sen, P., G. Namata, M. Bilgic and L. Getoor. 2008. Collective Classification in Network Data. *Technical Report* CS-TR-4905 and UMIACS-TR-2008-04.

Srikant, R and R. Agrawal. 1996. Mining sequential patterns: Generalization and performance improvements. *Advances in Database Technology*.

Su, Q., X. Xu, H. Guo, Z. Guo, X. Wu, X. Zhang, B. Swen, and Z. Su. 2008. Hidden sentiment association in chinese web opinion mining. *Proceedings of International Conference on World Wide Web*.

Toutanova, K., D. Klein, C. Manning, and Y. Singer. 2003. Feature-Rich Part-of-Speech Tagging with a Cyclic Dependency Network. *Proceedings of HLT-NAACL* 2003, pp. 252-259.

Wilson, T., J. Wiebe, and P. Hoffmann. 2005. Recognizing contextual polarity in phrase-level sentiment analysis. *Proceedings of the Human Language Technology Conference and the Conference on Empirical Methods in Natural Language Processing* (HLT/EMNLP-2005).

Wu, Y. and M. Wen. 2010. Disambiguating dynamic sentiment ambiguous adjectives. *Proceedings of the 23rd International Conference on Computational Linguistics* (Coling 2010).