# One Seed to Find Them All: Mining Opinion Features via Association

Zhen Hai, Kuiyu Chang, Gao Cong
School of Computer Engineering, Nanyang Technological University
50 Nanyang Avenue, Singapore 639798
{haiz0001, askychang, gaocong}@ntu.edu.sg

## ABSTRACT

Feature-based opinion analysis has attracted extensive attention recently. Identifying features associated with opinions expressed in reviews is essential for fine-grained opinion mining. One approach is to exploit the dependency relations that occur naturally between features and opinion words, and among features (or opinion words) themselves. In this paper, we propose a generalized approach to opinion feature extraction by incorporating robust statistical association analysis in a bootstrapping framework. The new approach starts with a small set of feature seeds, on which it iteratively enlarges by mining feature-opinion, feature-feature, and opinion-opinion dependency relations. Two association model types, namely likelihood ratio tests (LRT) and latent semantic analysis (LSA), are proposed for computing the pair-wise associations between terms (features or opinions). We accordingly propose two robust bootstrapping approaches, LRTBOOT and LSABOOT, both of which need just a handful of initial feature seeds to bootstrap opinion feature extraction. We benchmarked LRTBOOT and LSABOOT against existing approaches on a large number of real-life reviews crawled from the cellphone and hotel domains. Experimental results using varying number of feature seeds show that the proposed association-based bootstrapping approach significantly outperforms the competitors. In fact, one seed feature is all that is needed for LRTBOOT to significantly outperform the other methods. This seed feature can simply be the domain feature, e.g., "cellphone" or "hotel". The consequence of our discovery is far reaching: starting with just one feature seed, typically just the domain concept word, LRTBOOT can automatically extract a large set of high-quality opinion features from the corpus without any supervision or labeled features. This means that the automatic creation of a set of domain features is no longer a pipe dream!


*CIKM'12,* October 29–November 2, 2012, Maui, HI, USA.


## Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Information Filtering; I.2.7 [**Artificial Intelligence**]: Natural Language Processing—*Text Analysis*

## General Terms

Algorithms, Experimentation

## Keywords

opinion mining, sentiment analysis, aspect, feature, seed, bootstrapping, association

## 1. INTRODUCTION

Opinion mining, also known as sentiment analysis, is the computational study of subjectivity, i.e., the sentiments and opinions expressed in online review texts. Opinions expressed in reviews can be analyzed at different resolutions [6, 17, 24, 8, 25]. Document-level opinion mining identifies the overall opinion expressed in a review document, but often fails to associate specific opinions with different aspects of the commented subject (or entity). This problem also happens, though to a lesser extent, in sentence-level opinion mining, as shown in Example 1.1:

EXAMPLE 1.1. *"The exterior is very beautiful, also not expensive, though the battery is not durable, I still unequivocally recommend this cellphone!"*

Example 1.1 on a whole expresses a positive opinion on the cellphone, but contains conflicting opinions associated with different aspects of the cellphone. The opinion orientations for the "cellphone" itself and its "exterior" are positive, but the opinion polarity for the aspect of "battery" is negative. Generally such fine-grained opinions may very well tip the balance in purchase decisions. Consumers are usually not satisfied with just the overall opinion rating of a product, but are eager to find out why it receives the rating, that is, which positive or negative features contribute to the overall rating of the product. It is thus important to extract the specific opinionated aspects and associate them with the corresponding opinions.

An *opinion feature*, or *feature* in short, is defined as the object or aspect on which users have expressed their opinions. A feature is called an *explicit feature*, if it is explicitly mentioned in a sentence, typically as a noun or noun phrase. If it is not expressed explicitly but is implied, the feature is

called an *implicit feature*. Example 1.1 contains three explicit opinion features and one implicit feature. The explicit features, i.e., "exterior", "battery", and "cellphone" are associated with opinion words, "beautiful", "durable", and "recommend", respectively. The implicit feature "price" does not appear explicitly, but is implied in the opinion word "expensive". In this paper, we focus on identifying explicit opinion features from reviews.

Existing approaches to feature extraction can be classified roughly into supervised and unsupervised learning categories. Supervised learning approaches tend to yield relatively accurate results in a given domain, e.g., cellphone reviews, provided that a large set of annotated training data is available. However, the models must be retrained when applied to different domains such as hotel reviews [11, 14, 10]. Moreover, high-quality annotated training review data are hard to come by, which can be tedious and expensive to collect.

In contrast, unsupervised learning approaches have no such limitations, given the ready availability of unlabeled review data online. *Natural language processing* (NLP) methods attempt to recognize features by manually defining extraction rules or templates based on syntactic parsing [21, 19]. Though giving good domain independence, NLP methods do not work well with the informal writing styles used in online reviews, which often contain overly concise or grammatically incorrect sentences. One other problem associated with NLP methods is the limited coverage of the manually compiled syntactic rules.

Unsupervised corpus statistical approaches aim to extract opinion features by capturing their distributional characteristics from the corpus [26, 8, 18]. As a result, they are somewhat resistant to the colloquial nature of online reviews, provided that a suitably large corpus is used. However, the lack of ground-truth knowledge, such as annotated feature terms (i.e., seeds), frequently leads to incoherent and imprecise results. The results quickly deteriorate with every additional iteration, as the error accumulates, as in the case of blind leading the blind. In addition, unsupervised topic modeling approaches tend to discover coarse-grained and generic opinion topics or aspects, which are just clusters of related opinion features instead of the specific opinionated features themselves [7, 1, 22].

It is observed that co-occurrence dependency relations exist naturally between opinion words and features, even among features (opinion words) themselves. For example, an opinion word "expensive" is often used to modify the feature "price" in cellphone reviews. Such statistical associative relations between pair-wise terms (features or opinions) can be exploited to bootstrap opinion features based on a labeled set of feature seeds.

In this paper, we therefore propose a generalized bootstrapping framework to identify opinion features by employing term-to-term corpus statistics association analysis. The bootstrapping approach starts from a small manually prespecified list of feature seeds, which it then iteratively enlarges by mining pair-wise feature-opinion, feature-feature, and opinion-opinion associative relations between terms (features or opinion words) in the corpus. Two association model types are introduced in the bootstrapping framework, namely *likelihood ratio tests* (LRT) [4] and *latent semantic analysis* (LSA) [3], which we termed LRTBOOT and LSABOOT, respectively.

The proposed association-based bootstrapping framework has two advantages: (1) The statistical association analysis is more robust in mining dependency relations between pair-wise terms (features and opinion words), compared to syntactic parsing, and works especially well on real-life online text reviews; (2) The bootstrapping strategy achieves surprisingly good performance with only one word in the seed set. This single seed could simply be the domain concept term, e.g., "cellphone" for cellphone reviews and "hotel" for hotel reviews. Experiments conducted on real-life review domains demonstrate the effectiveness of our approach.

The remainder of this paper is structured as follows: In Section 2, we discuss related work. In Section 3, we describe the proposed corpus statistics association based bootstrapping approach to opinion feature extraction. We evaluate our association-based bootstrapping approach in Section 4 and conclude the paper with a brief discussion in Section 5.

## 2. RELATED WORK

In feature-based opinion mining, identifying a feature associated with an opinion is essential for fine-grained analysis of online reviews. Feature extraction is an ongoing research problem, with the vast majority of existing work done in the product review domain.

By formulating opinion mining as a joint structural tagging problem, supervised learning models including Hidden Markov Models and Conditional Random Fields have been used to tag features or aspects of commented entities [11, 14]. Supervised models may be carefully tuned to perform well on a given domain, but need extensive retraining when applied to a different domain, unless transfer learning is adopted [16].

Syntactic relationships between features and opinions can be naturally used to locate opinion features in an unsupervised manner. A syntactic parsing based *double propagation* (DP) approach was proposed to address the extraction of features as well as opinion words [19, 20]. DP recognizes the dependency relations between pair-wise terms (features and opinions) by manually defining eight syntactic rules. It then extracts features and opinion words iteratively using the extracted known features and opinions via the recognized syntactic relations. Independently, a similar method was proposed to extract opinionated features by defining a set of dependent syntactic rules between features and opinions in [21]. In addition, a semantic role labeling based approach was introduced to recognize opinion features by analyzing their semantic roles and local context in sentences of online news media texts [13].

Topic modeling approaches, such as *latent Dirichlet allocation* (LDA), which is a generative three-way (term-topic-document) probabilistic model [1], have been used to solve aspect-based opinion mining (or sentiment analysis) problems. The approaches actually tend to identify coarse-grained topics or aspects that correspond to distinguishing properties of the commented entities, which may not necessarily be opinionated product features, but rather user-defined clusters of specific opinion features [7, 1, 22, 15, 28]. For example, "where" could be a valid LDA topic-word associated with cellphone reviews, since users like to discuss about phone vendors, but it is not a product feature per se. In addition, the approach usually assumes that each review sentence contains exactly one topic [12], which is often violated in real-life online reviews.

An unsupervised corpus statistics approach was proposed to extract product features by using association rule mining [9]. The approach mines frequent itemsets as potential features, i.e., frequent nouns with high sentence frequencies (i.e., support). Pruning is necessary to remove many irrelevant features. The limitations of this approach lie in: (1) Frequent but invalid features are extracted incorrectly, and (2) rare but valid features may be missed out.

Our work aims to extract opinion features, which is related to the aforementioned existing work, but differs in three aspects. First, we use corpus statistics term-to-term associations to bootstrap feature terms, which performs better than simple syntactic parsing, especially on colloquial or informal review data. Second, in contrast with typical unsupervised statistics models, a manually pre-defined list of feature seeds is used. Third, the seed features are used not to train a model but as references for the iterative extraction process. In a way, our proposed statistical association based bootstrapping approach can be viewed as a semi-supervised approach. In the extreme case of using a single domain concept word (e.g., "cellphone" for cellphone review domain) as the sole seed feature, our approach even borders on the unsupervised.

# 3. OPINION FEATURE EXTRACTION

## 3.1 Overview

When users express their comments on a feature, a certain cluster of opinion words will be frequently used. For example, the feature term "price" is often associated with a cluster of opinion words like "expensive", "cheap", etc. Quite similarly, an opinion word usually covers a certain group of feature terms that are semantically related to each other. Semantic dependency relations thus exist naturally between opinion words and features in real-world reviews. In some instances, co-occurrence associations also exist among features (opinion words) themselves, since a user could express his or her opinions on several different product features in a single review. e.g., "screen" and "battery".

Such co-occurrence associative patterns between features and opinion words, as well as among features (or opinion words) themselves, can be measured and quantified. We therefore propose to exploit the distributional characteristics of opinion features in review corpus, with the goal of identifying them.

We define three types of term-to-term associations, i.e., *feature-opinion* (FO or OF), *feature-feature* (FF), and *opinion-opinion* (OO) to capture the aforementioned co-occurrence dependency relations in our opinion feature extraction task. Moreover, if opinion feature extraction uses only FO association, it could miss out a lot of related feature as well as opinion words. For example, given a known opinion word "expensive", we can extract a set of feature terms semantically related to "price" via the FO association analysis. However, we would miss out some valid co-occurring opinion words like "high" if we do not consider OO associations, which in return limits the feature extraction performance. Likewise, from the extracted known features like "price", we may only recognize semantically related opinion terms like "expensive" or "cheap" using the FO association. In this case, we would miss out valid features like "screen" that tend to co-occur with the features unless we consider FF associations. Thus in practice we need to consider FF and OO associations in addition to the FO association. This is a fact that is frequently overlooked in features identification analysis, since most existing research tends to focus on only the FO relationship.

We observed that candidate features strongly associated with invalid features (or opinions) tend to be non-features, while candidates strongly associated with valid features (opinions) are most likely true opinion features. Paraphrasing the age-old adage, "features of a feather flock together". In other words, without any ground truth in the form of known seed features, pair-wise term association based approaches may lead to too many frivolous features. Therefore, to extract opinion features reliably, we must start out with a manually crafted set of seed features, which is also known as the initial ground truth seed set. We then iteratively enlarge this feature set by adding newly identified candidate features that are statistically and strongly associated with a known member in the set.

The feature set is thus like an exclusive invitation-only finals club, where a new member (feature) is inducted into the club if and only if he or she is well-known (strongly associated) to any existing member (feature in the feature set). The initial club founders (seed features) thus play an essential role; They must "know" the most important future members (features) of the club, who would bring in yet more members (features).

Given a review corpus, our proposed approach initially extracts a set of candidate features and a set of candidate opinion words. It identifies two new sets of features and opinions that have strong FF and FO associations with the known set of features (a pre-defined list of feature seeds for the first step, and extracted known features thereafter), respectively. Based on the extracted known opinion set, the approach then identifies two sets of opinions and features via OO and FO association analysis, respectively. The feature identification process is performed iteratively until a suitable stopping criterion is met. Upon termination, we can finally bootstrap a validated set of opinion features (as well as an opinion word set).

We will describe our term-to-term association based bootstrapping algorithm in detail below.

## 3.2 Association-based Bootstrapping

Given a review corpus $\mathcal{C}$, we first need to generate two candidate sets of features and opinions, from which valid features as well as opinions are bootstrapped via corpus statistics association.

Opinion features typically occur as nouns (noun phrases), and tend to be the subject or object of a sentence. Simply selecting nouns or nouns phrases as feature candidates gives good coverage (recall), but comes at the expense of letting in too many noisy candidates, which may negatively impact the subsequent feature extraction process.

Using dependency parsing, we attempt to accurately generate a candidate feature set $\mathcal{CF}$ ($\mathcal{CF}$=$\{cf_1, \ldots, cf_i, \ldots, cf_M\}$, $M$: set size), which comprises only nouns (noun phrases) with "SBV" (subject-verb), "VOB" (verb-object), or "POB" (preposition-object) dependency relations in the corpus $\mathcal{C}$. According to experimental results, our candidate feature extraction yields respectable recall rates of 83.8% and 75.22% on the cellphone and hotel domains, respectively. Next we use all adjectives and verbs in the review corpus $\mathcal{C}$ to form

a candidate opinion word set $\mathcal{CO}$ ($\mathcal{CO}$=$\{co_1, \ldots, co_j, \ldots, co_N\}$, $N$: set size).

| Matrix | Entry | Association Type |
|---|---|---|
| $M_{FO}$ | $A(cf_i, co_j)$ | *feature-opinion* (FO) |
| $M_{FF}$ | $A(cf_{i_1}, cf_{i_2})$ | *feature-feature* (FF) |
| $M_{OO}$ | $A(co_{j_1}, co_{j_2})$ | *opinion-opinion* (OO) |

**Table 1: Association matrices for bootstrapping opinion feature extraction.**

We then compute a pair-wise term association matrix for each of the three aforementioned types of associative relations based on the set of candidate features as well as the candidate set of opinion words. As shown in Table 1, $M_{FO}$, $M_{FF}$, and $M_{OO}$ represent *feature-opinion* (FO or OF), *feature-feature* (FF), and *opinion-opinion* (OO) association matrices, respectively. Let $A$ denote the generalized corpus statistics association model used to compute the pair-wise term association. Accordingly, the $A(cf_i, co_j)$ ($cf_i \in \mathcal{CF}$, $co_j \in \mathcal{CO}$), $A(cf_{i_1}, cf_{i_2})$ ($cf_{i_1}$, $cf_{i_2} \in \mathcal{CF}$), and $A(co_{j_1}, co_{j_2})$ ($co_{j_1}$, $co_{j_2} \in \mathcal{CO}$) are FO, FF, and OO association scores estimated using the association model $A$ on the given review data $\mathcal{C}$.

Taking the *feature-opinion* association matrix $M_{FO}$ for instance, to calculate the pair-wise term association entry $A(cf_i, co_j)$ of the matrix, we first need to obtain the corpus occurrence statistics related to candidate feature $cf_i$ and candidate opinion $co_j$. We then estimate the association score between the candidates $cf_i$ and $co_j$ based on their corpus statistics by applying the association model $A$. We compute two other association matrices $M_{FF}$ and $M_{OO}$ in a similar manner.

Our generalized corpus statistics association based bootstrapping approach, ABOOT in short, is summarized in Algorithm 1. Some variables in the algorithm are defined as follows:

1. $\mathcal{S}$: a manually annotated feature seed set that is used to supervise the bootstrapping of feature extraction.
2. $\mathcal{F}$: a feature set that keeps track of the extracted features, initially $\mathcal{F} = \mathcal{S}$.
3. $\mathcal{O}$: opinion word set that tracks the extracted opinion words.
4. $A(t_1, t_2)$: an association score estimated via an association model $A$ for terms $t_1$ and $t_2$.
5. *foth*, *ffth*, and *ooth*: three thresholds for the FO (or OF), FF, and OO associations, respectively.

In Algorithm 1, We first extract two candidate sets of features and opinions in line 1 and line 2, respectively. The known feature set $\mathcal{F}$ is initialized with the annotated set of feature seeds in line 3. From line 6 to line 19, we bootstrap new features as well as opinion words, which have strong associations with the known or extracted features in the set $\mathcal{F}$. We then bootstrap more opinion words and features based on the extracted known opinion set $\mathcal{O}$ from line 20 to line 33. In line 34 of the algorithm, we update both known feature and opinion sets $\mathcal{F}$ and $\mathcal{S}$ with the extracted features and opinion words. The bootstrapping process is performed repeatedly from line 5, and will be terminated until no new opinion features are identified, as shown in line 35. Finally, a validated set of opinion features can be bootstrapped from the given review corpus $\mathcal{C}$.

**Algorithm 1** Corpus statistics association based bootstrapping algorithm.

**Require:** Review corpus $\mathcal{C}$, a labeled feature seed set $\mathcal{S}$
**Ensure:** A validated set of opinion features
1: $\mathcal{CF}$ ← Extract a candidate feature set from corpus $\mathcal{C}$;
2: $\mathcal{CO}$ ← Extract a candidate opinion set from corpus $\mathcal{C}$;
3: $\mathcal{F}$←$\mathcal{S}$;
4: $\mathcal{O}$←$\emptyset$;
5: **repeat**
6: **for** *each known feature* $f$ *in* $\mathcal{F}$ **do**
7: **for** *each candidate feature* $cf$ *in* $\mathcal{CF}$ **do**
8: **if** $(A(f, cf) \geq ffth)$ *AND* $(cf \notin \mathcal{F})$ **then**
9: Identify candidate $cf$ as a feature;
10: Remove candidate $cf$ from set $CF$;
11: **end if**
12: **end for**
13: **for** *each candidate opinion word* $co$ *in* $\mathcal{CO}$ **do**
14: **if** $(A(f, co) \geq foth)$ *AND* $(co \notin \mathcal{O})$ **then**
15: Identify candidate $co$ as an opinion word;
16: Remove candidate $co$ from set $CO$;
17: **end if**
18: **end for**
19: **end for**
20: **for** *each known opinion word* $o$ *in* $\mathcal{O}$ **do**
21: **for** *each candidate opinion word* $co$ *in* $\mathcal{CO}$ **do**
22: **if** $(A(o, co) \geq ooth)$ *AND* $(co \notin \mathcal{O})$ **then**
23: Identify candidate $co$ as an opinion word;
24: Remove candidate $co$ from set $CO$;
25: **end if**
26: **end for**
27: **for** *each candidate feature* $cf$ *in* $\mathcal{CF}$ **do**
28: **if** $(A(o, cf) \geq foth)$ *AND* $(cf \notin \mathcal{F})$ **then**
29: Identify candidate $cf$ as a feature;
30: Remove candidate $cf$ from set $CF$;
31: **end if**
32: **end for**
33: **end for**
34: Update $\mathcal{F}$ and $\mathcal{O}$ with identified features and opinions;
35: **until** *No new opinion features are identified*
36: **return** An identified opinion feature set $\mathcal{F}$

We illustrate Algorithm 1 using a straightforward example in Figure 1. Given a sample review corpus $\mathcal{C}$ (which contains 4 reviews only), We first extract two candidate feature and opinion sets $\mathcal{CF}$ and $\mathcal{CO}$ from the corpus $\mathcal{C}$. We then compute a table of pair-wise term association scores by employing an associative model $A$ based on the two candidate sets, as shown at the lower portion of the figure. Note that the association matrix (matrices) shown in the shaded grey area of the table corresponds to FO associative relation, and the matrices shown at the top left and bottom right of the table correspond to FF and OO association relations, respectively. Given a pre-specified threshold $thd$ =2.0 (which is used for FO, FF, and OO associations), by applying the ABOOT algorithm, we finally bootstrapped an extended feature set $\mathcal{F}$ as well as an opinion set $\mathcal{O}$, based on a manually annotated feature seed set $\mathcal{S}$ (which contains a noun feature "screen" only), as shown at the upper right portion of the figure.

Reviews:

1. The screen is really big, but the price is too expensive!
2. The price is expensive, students don't buy it usually.
3. The screen is beautiful, but the price is not!
4. The screen is big and beautiful!

*CF*={screen, price, student}
*CO*={big, expensive, buy, beautiful}

*S*={screen}
*thd*=2.0
*F*={screen, price}
*O*={big, beautiful, expensive}

| *A* | screen | price | student | big | expensive | buy | beautiful |
|---|---|---|---|---|---|---|---|
| screen | | 2.5 | 0.5 | 3.0 | 1.5 | 0.5 | 3.0 |
| price | 2.5 | | 1.0 | 1.5 | 3.0 | 1.5 | 1.5 |
| student | 0.5 | 1.0 | | 0.5 | 0.5 | 1.0 | 0.5 |
| big | 3.0 | 1.5 | 0.5 | | 2.0 | 0.5 | 2.0 |
| expensive | 1.5 | 3.0 | 0.5 | 2.0 | | 1.0 | 2.0 |
| buy | 0.5 | 1.5 | 1.0 | 0.5 | 1.0 | | 0.5 |
| beautiful | 3.0 | 1.5 | 0.5 | 2.0 | 2.0 | 0.5 | |

**Figure 1: A working example for the ABOOT algorithm.**

## 3.3 Association Models for Bootstrapping

As shown in Algorithm 1, the proposed generalized strategy for opinion feature extraction is called ABOOT (Association-based Bootstrapping). Different pair-wise term association measures can lead to different instance approaches.

There are two schools of thoughts on estimating pair-wise dependency relations: one is the tests for statistical significance, the other is the association measure. On the tests for statistical significance front, we choose the likelihood ratio tests model to estimate the pair-wise association for feature bootstrapping, which we call LRTBOOT (Likelihood Ratio Tests based Bootstrapping). For the association measure, we use latent semantic analysis as well as cosine similarity, which we call LSABOOT (Latent Semantic Analysis based Bootstrapping). Other types of association models can be evaluated as part of future work.

### 3.3.1 Likelihood Ratio Tests for Bootstrapping

We first describe the *likelihood ratio tests* (LRT) [4] association model. The LRT is well known for not relying critically on the assumption of normality, instead, it uses the asymptotic assumption of the generalized likelihood ratio. In practice, the use of likelihood ratios tends to result in significant improvements in text-analysis performance, even with relatively small amount of data [4].

LRT computes a contingency table of two term $T_i$ and $T_j$, derived from corpus statistics, as given in Table 2, where $k_1(T_i, T_j)$ is the number of documents (reviews) containing both terms $T_i$ and $T_j$; $k_2(T_i, \bar{T}_j)$ is the number of documents containing term $T_i$ but not $T_j$; $k_3(\bar{T}_i, T_j)$ is the number of documents containing term $T_j$ but not $T_i$; $k_4(\bar{T}_i, \bar{T}_j)$ is the number of documents containing neither $T_i$ nor $T_j$. Note that our purpose here is to measure how greatly pair-wise terms are associated with each other given the corpus statistics, rather than performing an actual statistics test.

| Corpus statistics | $T_j$ | $\bar{T}_j$ |
|---|---|---|
| $T_i$ | $k_1(T_i, T_j)$ | $k_2(T_i, \bar{T}_j)$ |
| $\bar{T}_i$ | $k_3(\bar{T}_i, T_j)$ | $k_4(\bar{T}_i, \bar{T}_j)$ |

**Table 2: Contingency table derived from corpus statistics.**

Based on the corpus statistics shown in Table 2, the *likelihood ratio tests* (LRT) [4] model captures the statistical association between terms $T_i$ and $T_j$ by employing the following function:

$$-2log\lambda = 2[logL(p_1, k_1, n_1) + logL(p_2, k_2, n_2) - logL(p, k_1, n_1) - logL(p, k_2, n_2)] \quad (1)$$

where,
$L(p, k, n)$=$p^k(1-p)^{n-k}$; $n_1$=$k_1 + k_3$; $n_2$=$k_2 + k_4$;
$p_1 = k_1/n_1$; $p_2$= $k_2/n_2$; $p = (k_1 + k_2)/(n_1 + n_2)$;

The higher the quantity $-2log\lambda$, the greater the statistical association between term $T_i$ and term $T_j$. We abbreviate this LRT based bootstrapping as LRTBOOT.

### 3.3.2 Latent Semantic Analysis for Bootstrapping

Generally, given a term-by-document matrix representing a collection of documents, *latent semantic analysis* (LSA) [3] applies *singular value decomposition* (SVD) to the matrix to statistically estimate the latent dimensions (or factors) and term-term associations of the collection.

In particular, we first build a term by document matrix $X$, given a corpus of review documents. By applying SVD, the term-by-document matrix is then decomposed into a product of three matrices:

$$X = LVR'$$

where $L$ and $R$ are the left and right singular matrices, and $V$ is a diagonal matrix of *singular values*.

Let $r$ be the rank of the raw matrix $X$. We select a value $k \ll r$. Let $V_k$ denote the diagonal matrix generated by choosing the top $k$ singular values from the matrix $V$, and let $L_k$ and $R_k$ be matrices generated by selecting the corresponding columns from the matrices $L$ and $R$, respectively. We thus obtain a reduced matrix $X_k$ by multiplying the three new matrices:

$$X_k = L_k V_k R'_k$$

The matrix $X_k$ is the best low rank ($k$) approximation to the raw matrix $X$, which minimizes the *Frobenius norm* [5] or reconstruction error in the form:

$$\| E \|_F = \sqrt{\sum_{t=1}^{T}\sum_{d=1}^{D} |e_{td}|^2}$$

where $E = X - X_k$, $e_{td}$: element of matrix $E$, $T$: term set size, and $D$: corpus size.

In the new latent space, we measure pair-wise term associations via cosine similarity of the corresponding row vectors of the "smoothed" matrix $X_k$. The LSA model is one type of the generalized association model $A$ used to compute the FF, FO, and OO term associations in Algorithm 1. We abbreviate this approach as LSABOOT.

## 4. EXPERIMENTS

We evaluate our new corpus statistics association based bootstrapping approach using a large number of real-life Chinese reviews on two different domains of cellphone and hotel. Note that the proposed approach is language independent and can be easily extended to different language based feature extraction applications.

## 4.1 Review Data

The cellphone review corpus comprises 7800 real-world reviews (or 12,564 sentences) collected from a major Chinese product website [1]. The hotel review corpus contains 4900 reviews (or 18,239 sentences) crawled from a famous Chinese travel portal [2]. All review documents in the two corpora were parsed using a Chinese language analysis tool named *Language Technology Platform* (LTP) [2].

To create a golden standard for the evaluation of feature extraction, 508 reviews were randomly selected from the cellphone review corpus and two annotators independently labeled opinion features. An annotated opinion feature is confirmed valid if it is marked by both annotators. If only one annotator marked an opinion feature, a third person made the final judgement. We obtained a total of 995 opinion features for the cellphone reviews. Similarly, we annotated 1013 opinion features from 206 randomly selected hotel reviews. The Kappa coefficients, a quantitative measure of inter-annotator agreement, are 0.66 and 0.62 on the cellphone and hotel reviews, respectively. Generally, a Kappa value in the range of 0.6-0.8 denotes substantial agreement [23].

## 4.2 Precision versus Recall

We first evaluate the feature extraction performance of the proposed LRTBOOT and LSABOOT against two state-of-art competitors, DPHITS and DP, and a BASELINE method.

The BASELINE approach simply evaluates precision and recall for the features in the seed feature set $\mathcal{S}$. DP (*double propagation*) extracts opinion features by using syntactic relations identified via manually defined dependency rules [19, 20]. Though giving good domain-independency, DP cannot effectively parse online real-world reviews, which are mostly informal and frequently contain grammatically incorrect sentences. As a result the defined syntactic rules have very limited coverage in real-world application. Notwithstanding, it is difficult to come up with a comprehensive set of dependency relations between features and opinions to cover all real-life cases. DPHITS uses *hyperlink-induced topic search* algorithm (HITS) to validate potential features recognized by DP plus two additional syntactic patterns of "part-whole" and "no" [27].

To be fair, the evaluation results for all methods used the same set of 10 seed features and the same review data partition. The choice of 10 seed features seems reasonable, as most people can easily come up with 10 features for a review domain.

### 4.2.1 Cellphone Reviews

In Table 3, we first list the best F-measure performance for the proposed association-based bootstrapping methods, LRTBOOT and LSABOOT, as well as three comparison methods, DPHITS, DP, and BASELINE, using 10 feature seeds on cellphone reviews. LRTBOOT achieved the best F-measure of 73.25%, which is 9.96%, 13.32%, and 22.66% better than that of DPHITS, DP, and BASELINE, respectively. LSABOOT had F-measure of 71.07%, which is 7.78%, 11.14%, and 20.48% better than DPHITS, DP, and BASELINE. The maximum recall for LSABOOT is 81.11%.

[1] product.tech.163.com

[2] www.lvping.com/hotels

| Method | Precision | Recall | F-measure |
|---|---|---|---|
| LRTBOOT | 68.00% | 79.40% | **73.25%** |
| LSABOOT | 63.25% | **81.11%** | 71.07% |
| DPHITS | 71.00% | 57.08% | 63.29% |
| DP | 70.42% | 52.16% | 59.93% |
| BASELINE | **95.80%** | 34.37% | 50.59% |

**Table 3: Best feature extraction F-measure on cellphone reviews.**

BASELINE has the largest precision of 95.80% but gives the lowest recall of 34.37%. This is expected since it selects only the top 10 seeds as the identified features. Dependency parsing rule based methods like DP tend to perform badly on real-life review corpus that contains large amounts of informal language or grammatically incorrect content. By employing the *hyperlink-induced topic search* (HITS) algorithm to filter potential features extracted by DP, in addition to engaging two new syntactic dependency patterns, DPHITS improves the feature extraction performance compared to DP. However, it still performed worse than LRTBOOT and LSABOOT.

We then plot precision versus recall curves at various parameter settings for LRTBOOT and LSABOOT, as well as the three competing methods, as shown in Figure 2. Note that the precision-recall curve for DPHITS terminated early at recall levels less than 60%. For DP and BASELINE, only one precision-recall point is obtained each, as indicated by the "diamond" and "circle", respectively.

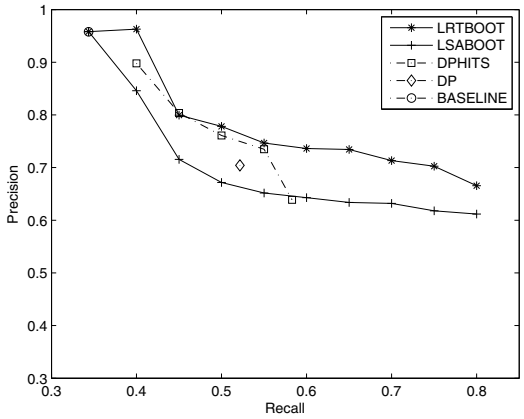


**Figure 2: Cellphone Feature extraction performance.**

From Figure 2, we see that the LRTBOOT curve lies well above those of DPHITS, DP, and BASELINE. Though starting out at a similar precision as LRTBOOT (as well as BASELINE) at the recall rate of approximately 35%, LSABOOT performed worse than LRTBOOT at increasing recall levels. At low recall levels, nearly all the methods perform well, achieving high precision.

Note that BASELINE achieved very high precision of 95.80% at its only precision-recall pair indicated by the "circle", which also happens to coincide with that of LRTBOOT and LSABOOT. DP attained a 70.42% precision at its unique recall of about 50%, which is 7.39% worse than LRTBOOT, but 3.25% better than LSABOOT. Across the recall levels from 40% to 60%, the mean precision of DPHITS is 76.72%

which is 3.75% lower than LRTBOOT, but 6.16% better than LSABOOT. This is not a serious problem considering that both the proposed LSABOOT and LRTBOOT can achieve much better coverage (higher recall) compared to other methods which stops at recall levels of less than 60%. In practice, a high recall and precision is more desirable.

Though giving good precision, BASELINE leads to bad coverage in the form of the low 35% recall. DP (the point indicated by a "diamond") has relatively better coverage/recall due to its eight manually crafted syntactic rules compared to BASELINE, but still performed worse than LRTBOOT and LSABOOT. This is expected due to the mismatch between DP rules and informal language used in online reviews. DPHITS is only marginally better than DP due to its use of HITS algorithm as well as two additional syntactic patterns. However, its coverage is still limited to at most 58.29% recall, with a corresponding 63.88% precision.

Based on the experiment results, LSABOOT did not achieve respectable precision values, though it outperformed the 3 other competitors in terms of having the highest recall rates of up to 81.11%. Thanks to the high recall rates, LSABOOT was able to achieve the overall second best F-measure of 71.07%, which is 7.78% better than the next competitor, DPHITS. LRTBOOT is the overall winner in terms of robustness and absolute F-measure. It was the de facto precision leader at all levels of recalls, and was able to achieve a recall of about 80%, at which its best F-measure of 73.25% was also achieved.

### 4.2.2 Hotel Reviews

The best results in F-measure for the two proposed approaches, LRTBOOT and LSABOOT, as well as the 3 benchmark methods of DPHITS, DP, and BASELINE, on hotel reviews are shown in Table 4.

| Method | Precision | Recall | F-measure |
|---|---|---|---|
| LRTBOOT | 57.43% | 67.13% | **61.90%** |
| LSABOOT | 45.34% | **69.60%** | 54.91% |
| DPHITS | 62.03% | 52.42% | 56.82% |
| DP | 55.69% | 50.74% | 53.10% |
| BASELINE | **92.67%** | 31.19% | 46.68% |

**Table 4: Best feature extraction F-measure on hotel reviews.**

Even though having the largest precision of 92.67%, BASELINE obtained the worst recall of 31.19%. LRTBOOT again obtained the best F-measure of 61.90%, which is 5.08%, 8.80%, and 15.22% better than that of DPHITS, DP, and BASELINE, respectively, LSABOOT achieved its best F-measure of 54.91% (at 69.60% recall level), which is 1.81% and 8.23% better than that of DP and BASELINE respectively, but 1.91% worse than DPHITS. However, LSABOOT still achieved the overall best recall of 69.60%.

Figure 3 plots the precision-recall performance curves for all methods, LRTBOOT, LSABOOT, DPHITS, DP, and BASELINE, on hotel reviews. Similarly to the cellphone results, LRTBOOT lies far above that of competitors for all recall levels, while LSABOOT is way inferior to LRTBOOT. At about 30% recall, the BASELINE precision is 92.67%, which is the same as that of both LRTBOOT and LSABOOT. DP achieved a 55.69% precision at its unique precision-recall point of approximately 50% recall. This is 15.13% lower than LRTBOOT, but 8.51% better than

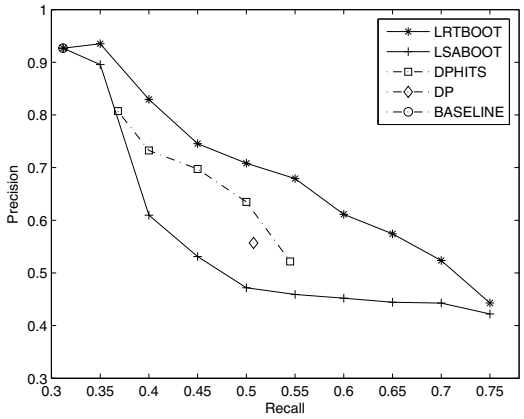


**Figure 3: Hotel Feature extraction performance.**

LSABOOT. Spanning the recall levels from 35% to 55%, the mean precision of DPHITS is around 67.87%, which is 10.08% worse than that of LRTBOOT, but 8.52% better than the mean LSABOOT precision.

Different from cellphone reviews, hotel reviews contain a large number of complex and noisy sentences, including a large number of irrelevant personal anecdotes or stories. This basically makes the exploration of latent (high-level) factors or structures of review data by LSA more challenging, and thus leads to relatively poor feature extraction performance of LSABOOT.

Based on the experimental results obtained on the cellphone and hotel review domains, we therefore conclude that the proposed statistical association based bootstrapping approach, especially LRTBOOT, gives significant performance improvement for opinion feature extraction, compared to the dependency parsing rule based methods of DP and DPHITS, as well as a dictionary (seed set) based BASELINE. Given a suitably large review corpus, statistical association analysis based bootstrapping method like LRTBOOT can more robustly discover the co-occurrence dependency relationships between pair-wise terms (features and opinions), and thus better discriminate valid features from the invalid ones, especially on real-life raw reviews that typically contain grammatically incorrect sentences or informal language.

## 4.3 Performance versus Seed Size

We then study the effect of seed set size on feature extraction for both LRTBOOT and LSABOOT, as well as DPHITS, DP, and BASELINE on cellphone and hotel reviews, respectively. To collect seed features, we simply rank all feature candidates by descending document frequency for each review domain. We manually selected up to the top 50 truthful features as seeds.

### 4.3.1 Cellphone Reviews

Figure 4 plots the feature extraction performance in F-measure versus top $K$ seeds (50 seeds max) for LRTBOOT, LSABOOT, and the three benchmark methods on cellphone reviews. Clearly, both LRTBOOT and LSABOOT outperformed the competitors for all seed sizes from 1 to 50. In particular, the mean F-measure of LRTBOOT across all observations is 73.62%, which is 10.35%, 14.07%, and 21.13% better than that of DPHITS, DP, and BASELINE, respec-

tively. The mean F-measure of LSABOOT is 71.02%, which is 7.75%, 11.47%, and 18.53% better than that of DPHITS, DP, and BASELINE, respectively.

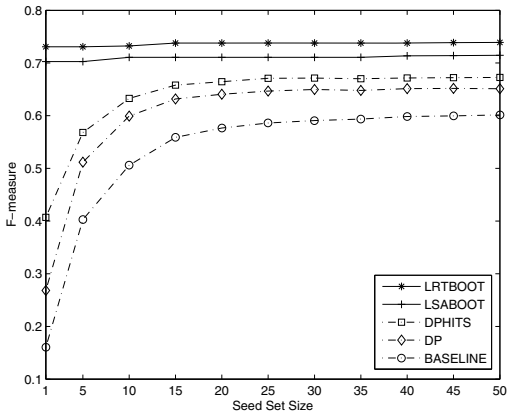


**Figure 4: F-measure versus seed size for cellphone reviews.**

Furthermore, when increasing seeds from 1 to 50, the F-measure for LRTBOOT remained almost constant, i.e., it does not depend critically on the seed set size. Remarkably, LRTBOOT achieved excellent performance with only one seed word, i.e., the domain concept term "cellphone"! A similar situation can also be observed for LSABOOT for cellphone reviews. This is because both LRTBOOT and LSABOOT bootstrap opinion features by relying on corpus statistics association analysis. In other words, given suitable association thresholds, the candidates that have relatively strong associations with the domain concept word (top 1 seed) or extracted known features (opinions) can be identified after several bootstrapping iterations. Therefore, we can always bootstrap a great many features via robust statistical association analysis based on a properly-sized seed set of 20 seeds or less. Moreover, a seed set size of one will also work as well as a seed size of 20!

The performance for DPHITS, DP, and BASELINE increases significantly over the lower spectrum of seed sizes, and begin to level-off at around 15 seeds. Clearly, seed set size has a big effect on the performance of DPHITS, DP, and BASELINE for feature extraction. This agrees well with the observation that dependency parsing based methods tend to suffer from the feature extraction coverage problem for their manually defined syntactic rules. On the other hand, their performance curves are in line with expectation, since the number of product features is finite, that is, when users comment on products, the feature vocabulary they use will converge [8]. Additionally, our feature seeds were hand-picked from the topmost frequent nouns (noun phrases). Thus we see a miniature version of Zipf's and Heap's Law at work here. Therefore, in order to achieve a decent performance for DPHITS, DP, and BASELINE, a relatively large seed set is needed in practice. In contrast, just one domain word suffices for our proposed LRTBOOT or LSABOOT.

### *4.3.2 Hotel Reviews*

We evaluate the performance of LRTBOOT and LSABOOT versus seed size on the domain of hotel reviews. Figure 5 plots the F-measure performance for LRTBOOT and LSABOOT, as well as DPHITS, DP, and BASELINE.

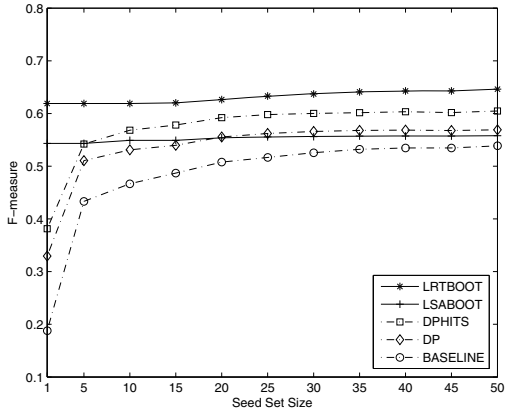


**Figure 5: F-measure versus seed size for hotel reviews.**

LRTBOOT still gives superior performance, and this time round LRTBOOT enjoys a slight visible improvement from 15 to 50 seed words. Again the LRTBOOT curve lies well above that of DPHITS, DP, and BASELINE for all seed sizes. The average F-measure of LRTBOOT across all observations is 63.15%, which is 6.14%, 9.81%, and 15.28% better than that of DPHITS, DP, and BASELINE, respectively. Compared to the BASELINE, LSABOOT showed better performance for all seed sizes, however it still underperformed LRTBOOT. LSABOOT outperformed DPHITS for 5 or less seeds, but did worse when seed size increased from 5 to 50. Similarly, compared to DP, though LSABOOT achieved better performance with less than 20 seeds, its curve lied under that of DP for seed sizes 20 to 50. The average F-measure of LSABOOT across all seed sizes is 55.27%, which is 1.93% and 7.40% better than DP and BASELINE, but 1.74% lower than that of DPHITS.

Moreover, the performance curves of both LRTBOOT and LSABOOT still exhibit little variation across all seed numbers on the hotel reviews. The results again validated our observation on the cellphone reviews. We hence conclude that seed set size has no large performance influence on our proposed bootstrapping approach. In fact we can again select just the domain concept word to apply LRTBOOT as well as LSABOOT to real-life reviews and achieve great feature extraction results with no supervision at all.

## 4.4 Association Thresholds

The choice of thresholds for FO, FF, and OO associations is very important for our proposed method to work in practice. In this section, we give evaluations on the performance (in F-measure) versus the thresholds, in an attempt to shine some light on this issue.

Figure 6 plots the LRTBOOT feature extraction performance versus FO association threshold, while keeping both FF and OO thresholds at 21.0 and 12.0 on the cellphone reviews, and 21.0 and 43.0 on the hotel reviews, respectively (note that FF and OO association thresholds are obtained experimentally). Similarly, we can also plot the performance versus FF or OO association threshold on the two domains.

In Figure 6, The performance of LRTBOOT initially in-

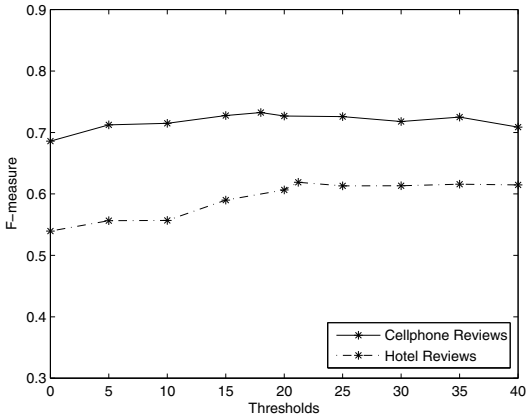


**Figure 6: LRTBOOT F-measure vs FO association threshold on cellphone and hotel domains.**

creased with the FO threshold, achieving the best F-measure of 73.25% and 61.90% at the FO thresholds of 18.0 and 21.0 on the cellphone and hotel reviews, respectively, and declined slightly thereafter. This is expected as a higher FO threshold will weed out the insignificant or noisy associations, which gives higher-quality features, However, increasing it beyond a certain point will kill some legitimate associations as well.

As shown in Figure 7, similar trends are observed in the performance curves of F-measure versus FO thresholds for LSABOOT, with both FF and OO thresholds at 0.54 and 0.60 on the cellphone, and 0.51 and 0.55 on the hotel domains. The LSABOOT curve grows gradually with the FO threshold, and subsequently achieved the best F-measure of 71.07% and 54.91% at the thresholds of 0.53 and 0.52 on the cellphone and hotel reviews, respectively. If we raise the threshold further, the curves will be further degraded.

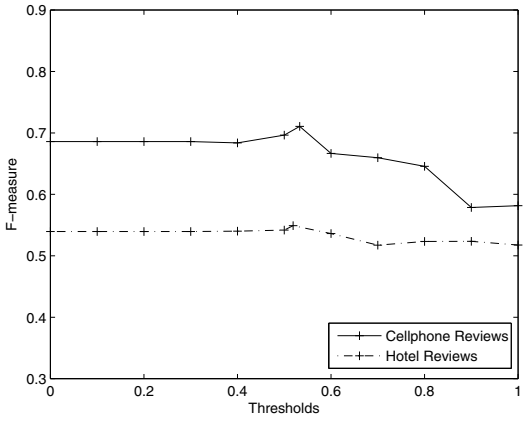


**Figure 7: LSABOOT feature extraction F-measure vs. FO association threshold on cellphone and hotel reviews.**

Association thresholds for our proposed statistical association based bootstrapping approach must be set. We currently use a standard grid-search procedure to find suitable threshold values that produce good opinion feature extraction performance. For instance, the FO threshold scores for both LRTBOOT and LSABOOT, in our case, can be selected near 20.0 and 0.5, respectively. In practice, suitable thresholds can be determined experimentally via cross-validation on a labeled dataset, which renders our approach a semi-supervised one.

# 5. CONCLUSIONS

In this paper, we propose a generalized corpus statistics association based bootstrapping approach for opinion feature extraction. Starting from a small manually labeled set of feature seeds, LRTBOOT as well as LSABOOT can bootstrap a large number of valid features by mining pairwise term associations via an effective statistical association model. Experimental results using varying numbers of seeds on the cellphone and hotel review domains demonstrate improvements of the proposed association-based approaches of LRTBOOT and LSABOOT over two state-of-art methods and one baseline method. In fact, LRTBOOT, as well as LSABOOT, achieved quite good performance with only one seed word, which is simply the domain concept word. This makes our proposed statistical association based bootstrapping approach, especially LRTBOOT, powerful and effective for practical opinion feature extraction in opinion mining and sentiment analysis.

Several weaknesses still exist:
(1) As a corpus statistics approach, the proposed approach is less successful in dealing with infrequent feature extraction. For instance, the infrequent feature "FM" (a slang for radio in cellphone reviews) is missed out. Basically, it is not a serious problem, since the number of infrequent features, compared to the number of frequent features, is minor. Moreover, frequent features are basically more important than infrequent ones [8].
(2) Our new approach does not currently extract non-noun opinion features, e.g., design, decorate.
(3) Our approach is at the mercy of errors in the POS-tagging and syntactic parsing for candidate feature extraction.

For future work, we will employ more advanced techniques like fine-grained topical modeling to jointly identify opinion features, including non-noun features, infrequent features, and implicit features as well. We also plan to further test the extracted opinion features in a real opinion mining system by giving every product a feature-specific sentiment summary.

## Acknowledgements

This research was supported in part by Singapore Ministry of Education's Academic Research Fund Tier 2 grant ARC 9/12 (MOE2011-T2-2-056).

# 6. REFERENCES

[1] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[2] W. Che, Z. Li, and T. Liu. Ltp: a chinese language technology platform. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 13–16, 2010.

[3] S. Deerwester, S. T. Dumais, and R. Harshman. Indexing by latent semantic analysis. *Journal of the American Society for Information Science and Technology*, 41:391–407, 1990.

[4] T. Dunning. Accurate methods for the statistics of surprise and coincidence. *Computational Linguistics*, 19(1):61–74, 1993.

[5] G. Golub and C. Van Loan. Matrix computations. *The Johns Hopkins University Press*, 3rd edition, 1996.

[6] V. Hatzivassiloglou and J. M. Wiebe. Effects of adjective orientation and gradability on sentence subjectivity. In *Proceedings of the 18th conference on Computational linguistics - Volume 1*, pages 299–305, 2000.

[7] T. Hofmann. Unsupervised learning by probabilistic latent semantic analysis. *Machine Learning Journal*, 42:177–196, 2001.

[8] M. Hu and B. Liu. Mining and summarizing customer reviews. In *Proceedings of the 10th ACM SIGKDD international conference on Knowledge Discovery and Data Mining*, pages 342–351, 2004.

[9] M. Hu and B. Liu. Mining opinion features in customer reviews. In *Proceedings of the 19th national conference on Artificial Intelligence*, pages 755–760, 2004.

[10] N. Jakob and I. Gurevych. Extracting opinion targets in a single- and cross-domain setting with conditional random fields. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1035–1045, 2010.

[11] W. Jin and H. H. Ho. A novel lexicalized hmm-based learning framework for web opinion mining. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 465–472, 2009.

[12] Y. Jo and A. H. Oh. Aspect and sentiment unification model for online review analysis. In *Proceedings of the 4th ACM international conference on Web Search and Data Mining*, pages 815–824, 2011.

[13] S.-M. Kim and E. Hovy. Extracting opinions, opinion holders, and topics expressed in online news media text. In *Proceedings of the Workshop on Sentiment and Subjectivity in Text*, pages 1–8, 2006.

[14] F. Li, C. Han, M. Huang, X. Zhu, Y.-J. Xia, S. Zhang, and H. Yu. Structure-aware review mining and summarization. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 653–661, 2010.

[15] C. Lin and Y. He. Joint sentiment/topic model for sentiment analysis. In *Proceeding of the 18th ACM conference on Information and Knowledge Management*, pages 375–384, 2009.

[16] S. J. Pan and Q. Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, 2010.

[17] 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*, pages 79–86, 2002.

[18] A. Popescu and O. Etzioni. Extracting product features and opinions from reviews. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing (HLT/EMNLP)*, pages 339–346, 2005.

[19] G. Qiu, B. Liu, J. Bu, and C. Chen. Expanding domain sentiment lexicon through double propagation. In *Proceedings of the 21st international jont conference on Artificial Intelligence*, pages 1199–1204, 2009.

[20] G. Qiu, B. Liu, J. Bu, and C. Chen. Opinion word expansion and target extraction through double propagation. *Computational Linguistics*, 37:9–27, 2011.

[21] G. Qiu, C. Wang, J. Bu, K. Liu, and C. Chen. Incorporate the syntactic knowledge in opinion mining in user-generated content. In *Proceedings of the NLPIX'08 Workshop*, 2008.

[22] I. Titov and R. McDonald. Modeling online reviews with multi-grain topic models. In *Proceeding of the 17th international conference on World Wide Web*, pages 111–120, 2008.

[23] A. J. Viera and J. M. Garrett. Understanding interobserver agreement: the kappa statistic. *Family Medicine*, 37(5):360–363, 2005.

[24] J. Wiebe, T. Wilson, R. Bruce, M. Bell, and M. Martin. Learning subjective language. *Computational Linguistics*, 30(3):277–308, 2004.

[25] T. Wilson, J. Wiebe, and P. Hoffmann. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 347–354, 2005.

[26] J. Yi, T. Nasukawa, R. Bunescu, and W. Niblack. Sentiment analyzer: Extracting sentiments about a given topic using natural language processing techniques. In *Proceedings of the 3rd IEEE International Conference on Data Mining*, pages 427, 2003.

[27] L. Zhang, B. Liu, S. H. Lim, and E. O'Brien-Strain. Extracting and ranking product features in opinion documents. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, pages 1462–1470, 2010.

[28] W. X. Zhao, J. Jiang, H. Yan, and X. Li. Jointly modeling aspects and opinions with a maxent-lda hybrid. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 56–65, 2010.