Clustering for Simultaneous Extraction of Aspects and Features from Reviews

Lu Chen[1]*, Justin Martineau[2], Doreen Cheng[2], Amit Sheth[1]
[1] Kno.e.sis Center, Wright State University
Fairborn, OH 45435 USA
[2] Samsung Research America, Silicon Valley
San Jose, CA 95134 USA
{chen, amit}@knoesis.org
justin.m@samsung.com
doreenycheng@yahoo.com

Abstract

This paper presents a clustering approach that simultaneously identifies product features and groups them into aspect categories from online reviews. Unlike prior approaches that first extract features and then group them into categories, the proposed approach combines feature and aspect discovery instead of chaining them. In addition, prior work on feature extraction tends to require seed terms and focus on identifying explicit features, while the proposed approach extracts both *explicit* and *implicit* features, and does not require seed terms. We evaluate this approach on reviews from three domains. The results show that it outperforms several state-of-the-art methods on both tasks across all three domains.

1 Introduction

If you are thinking of buying a TV for watching football, you might go to websites such as Amazon to read customer reviews on TV products. However, there are many products and each of them may have hundreds of reviews. It would be helpful to have an aspect-based sentiment summarization for each product. Based on other customers' opinions on multiple aspects such as size, picture quality, motion-smoothing, and sound quality, you might be able to make the decision without going through all the reviews. To support such summarization, it is essential to have an algorithm that extracts product features and aspects from reviews.

Features are components and attributes of a product. A feature can be directly mentioned as an opinion target (i.e., explicit feature) or implied by opinion words (i.e., implicit feature). Different feature expressions may be used to describe the same *aspect* of a product. Aspect can be represented as a group of features. Consider the following review sentences, in which we denote **explicit** and *implicit* features in boldface and italics, respectively.

1. This phone has great **display** and perfect **size**. It's very *fast* with all great **features**.
2. Good **features** for an *inexpensive* android. The **screen** is *big* and *vibrant*. Great **speed** makes *smooth* viewing of tv programs or sports.
3. The phone runs *fast* and *smooth*, and has great **price**.

In review 1, *display* is an explicit feature, and opinion word "fast" implies implicit feature *speed*. The task is to identify both explicit and implicit features, and group them into aspects, e.g., {speed, fast, smooth}, {size, big}, {price, inexpensive}.

Many existing studies (Hu and Liu, 2004; Su et al., 2006; Qiu et al., 2009; Hai et al., 2012; Xu et al., 2013) have focused on extracting features without grouping them into aspects. Some methods have been proposed to group features given that feature expressions have been identified beforehand (Zhai et al., 2010; Moghaddam and Ester, 2011; Zhao et al., 2014), or can be learned from semi-structured Pros and Cons reviews (Guo et al., 2009; Yu et al., 2011). In recent years, topic models have been widely studied for their use in aspect discovery with the advantage that they extract features and group them simultaneously. However, researchers have found some

*This author's research was done during an internship with Samsung Research America.

Proceedings of NAACL-HLT 2016, pages 789–799,
San Diego, California, June 12-17, 2016.

limitations of such methods, e.g., the produced topics may not be coherent or directly interpretable as aspects (Chen et al., 2013; Bancken et al., 2014), the extracted aspects are not fine-grained (Zhang and Liu, 2014), and it is ineffective when dealing with aspect sparsity (Xu et al., 2014).

In this paper, we present a clustering algorithm that extracts features and groups them into aspects from product reviews. Our work differs from prior studies in three ways. First, it identifies both features and aspects simultaneously. Existing clustering-based solutions (Su et al., 2008; Lu et al., 2009; Bancken et al., 2014) take a two-step approach that first identifies features and then employs standard clustering algorithms (e.g., k-means) to group features into aspects. We propose that these two steps can be combined into a single clustering process, in which different words describing the same aspect can be automatically grouped into one cluster, and features and aspects can be identified at the same time. Second, both explicit and implicit features are extracted and grouped into aspects. While most existing methods deal with explicit features (e.g., "speed", "size"), much less effort has been made to identify implicit features implied by opinion words (e.g., "fast", "big"), which is challenging because many general opinion words such as "good" or "great" do not indicate product features, therefore they should not be identified as features or grouped into aspects. Third, it is unsupervised and does not require seed terms, hand-crafted patterns, or any other labeling efforts.

Specifically, the algorithm takes a set of reviews on a product (e.g., TV, cell phone) as input and produces aspect clusters as output. It first uses a part-of-speech tagger to identify nouns/noun phrases, verbs and adjectives as candidates. Instead of applying the clustering algorithm to all the candidates, only the frequent ones are clustered to generate *seed clusters*, and then the remaining candidates are placed into the closest seed clusters. This does not only speed up the algorithm, but it also reduces the noise that might be introduced by infrequent terms in the clustering process. We propose a novel *domain-specific similarity measure* incorporating both statistical association and semantic similarity between a pair of candidates, which recognizes features referring to the same aspects in a particular domain. To further improve the quality of clusters, several problem-specific *merging constraints* are used to prevent the clusters referring to different aspects from being merged during the clustering process. The algorithm stops when it cannot find another pair of clusters satisfying these constraints.

This algorithm is evaluated on reviews from three domains: cell phone, TV and GPS. Its effectiveness is demonstrated through comparison with several state-of-the-art methods on both tasks of feature extraction and aspect discovery. Experimental results show that our method consistently yields better results than these existing methods on both tasks across all the domains.

2 Related Work

Feature and aspect extraction is a core component of aspect-based opinion mining systems. Zhang and Liu (2014) provide a broad overview of the tasks and the current state-of-the-art techniques.

Feature identification has been explored in many studies. Most methods focus on explicit features, including unsupervised methods that utilize association rules (Hu and Liu, 2004; Liu et al., 2005), dependency relations (Qiu et al., 2009; Xu et al., 2013), or statistical associations (Hai et al., 2012) between features and opinion words, and supervised ones that treat it as a sequence labeling problem and apply Hidden Markov Model (HMM) or Conditional Random Fields (CRF) (Jin et al., 2009; Yang and Cardie, 2013) to it. A few methods have been proposed to identify implicit features, e.g., using co-occurrence associations between implicit and explicit features (Su et al., 2006; Hai et al., 2011; Zhang and Zhu, 2013), or leveraging lexical relations of words in dictionaries (Fei et al., 2012). Many of these techniques require seed terms, hand-crafted rules/patterns, or other annotation efforts.

Some studies have focused on grouping features and assumed that features have been extracted beforehand or can be extracted from semi-structured Pros and Cons reviews. Methods including similarity matching (Carenini et al., 2005), topic modeling (Guo et al., 2009; Moghaddam and Ester, 2011), Expectation-Maximization (EM) based semi-supervised learning (Zhai et al., 2010; Zhai et al., 2011), and synonym clustering (Yu et al., 2011) have

been explored in this context.

To extract features and aspects at the same time, topic model-based approaches have been explored by a large number of studies in recent years. Standard topic modeling methods such as pLSA (Hofmann, 2001) and LDA (Blei et al., 2003) are extended to suit the peculiarities of the problem, e.g., capturing local topics corresponding to ratable aspects (Titov and McDonald, 2008a; Titov and McDonald, 2008b; Brody and Elhadad, 2010), jointly extracting both topic/aspect and sentiment (Lin and He, 2009; Jo and Oh, 2011; Kim et al., 2013; Wang and Ester, 2014), incorporating prior knowledge to generate coherent aspects (Mukherjee and Liu, 2012; Chen et al., 2013; Chen et al., 2014), etc.

Very limited research has focused on exploring clustering-based solutions. Su et al. (2008) presented a clustering method that utilizes the mutual reinforcement associations between features and opinion words. It employs standard clustering algorithms such as k-means to iteratively group feature words and opinion words separately. The similarity between two feature words (or two opinion words) is determined by a linear combination of their intra-similarity and inter-similarity. Intra-similarity is the traditional similarity, and inter-similarity is calculated based on the degree of association between feature words and opinion words. To calculate the inter-similarity, a feature word (or an opinion word) is represented as a vector where each element is the co-occurrence frequency between that word and opinion words (or feature words) in sentences. Then the similarity between two items is calculated by cosine similarity of two vectors. In each iteration, the clustering results of one type of data items (feature words or opinion words) are used to update the pairwise similarity of the other type of items. After clustering, the strongest links between features and opinion words form the aspect groups. Mauge et al. (2012) first trained a maximum-entropy classifier to predict the probability that two feature expressions are synonyms, then construct a graph based on the prediction results and employ greedy agglomerative clustering to partition the graph to clusters. Bancken et al. (2014) used k-medoids clustering algorithm with a WordNet-based similarity metric to cluster semantically similar aspect mentions.

These existing clustering methods take two steps. In the first step, features are extracted based on association rules or dependency patterns, and in the second step features are grouped into aspects using clustering algorithms. In contrast, our method extracts features and groups them at the same time. Moreover, most of these methods extract and group only explicit features, while our method deals with both explicit and implicit features. The method proposed in (Su et al., 2008) also handles implicit features (opinion words), but their similarity measure largely depends on co-occurrence between features and opinion words, which may not be efficient in identifying features that are semantically similar but rarely co-occur in reviews.

3 The Proposed Approach

Let $X = \{x_1, x_2, ..., x_n\}$ be a set of candidate features extracted from reviews of a given product (e.g., TV, cell phone). Specifically, by using a part-of-speech tagger[1], nouns (e.g., "battery") and two consecutive nouns (e.g., "battery life") are identified as candidates of *explicit* features, and adjectives and verbs are identified as candidates of *implicit* features. Stop words are removed from X. The algorithm aims to group similar candidate terms so that the terms referring to the same aspect are put into one cluster. At last, the important aspects are selected from the resulting clusters, and the candidates contained in these aspects are identified as features.

3.1 A Clustering Framework

Algorithm 1 illustrates the clustering process. The algorithm takes as input a set X that contains n candidate terms, a natural number k indicating the number of aspects, a natural number s ($0 < s \leq n$) indicating the number of candidates that will be grouped first to generate the *seed clusters*, and a real number δ indicating the upper bound of the distance between two mergeable clusters. Instead of applying the *agglomerative clustering* to all the candidates, it first selects a set $X' \subseteq X$ of s candidates that appear most frequently in the corpus for clustering. The reasons for this are two-fold. First, the frequently mentioned terms are more likely the actual features of customers' interests. By clustering these terms first, we can generate high quality seed clusters.

[1] http://nlp.stanford.edu/software/tagger.shtml

Second, as the clustering algorithm requires pairwise distances between candidates/clusters, it could be very time-consuming if there are a large number of candidates. We can speed up the process by clustering only the most frequent ones.

Algorithm 1: Clustering for Aspect Discovery

Input: $X = \{x_1, ..., x_n\}$, k, s, δ
Output: $\{A_j\}_{j=1}^{k}$

Select the top s most frequent candidates from X: $X' = \{x'_1, ..., x'_s\}$;
Set $C_1 \leftarrow \{x'_1\}, ..., C_s \leftarrow \{x'_s\}$;
Set $\Theta \leftarrow \{C_1, ..., C_s\}$;
while *there exist a pair of mergeable clusters from* Θ **do**
    Select a pair of closest clusters C_l and C_m such that VIOLATE-CONSTRAINTS(C_l, C_m, δ) is false;
    $C_m \leftarrow C_l \cup C_m$;
    $\Theta \leftarrow \Theta - \{C_l\}$;
for $x_i \in X - X'$ **do**
    Select the closest clusters C_d from Θ such that VIOLATE-CONSTRAINTS($\{x_i\}$, C_d, δ) is false;
    if *there exist such cluster* C_d **then**
        $C_d \leftarrow C_d \cup \{x_i\}$;
$\{A_j\}_{j=1}^{k} \leftarrow$ SELECT(Θ, k);

The clustering starts with every frequent term x'_i in its own cluster C_i, and Θ is the set of all clusters. In each iteration, a pair of clusters C_l and C_m that are most likely composed of features referring to the same aspect are merged into one. Both a similarity measure and a set of constraints are used to select such pair of clusters. We propose a *domain-specific similarity measure* that determines how similar the members in two clusters are regarding the particular domain/product. Moreover, we add a set of *merging constraints* to further ensure that the terms from different aspects would not be merged. The clustering process stops when it cannot find another pair of clusters that satisfy the constraints. We call the obtained clusters in Θ the *seed clusters*. Next, the algorithm assigns each of the remaining candidate $x_i \in X - X'$ to its closest seed cluster that satisfies the merging constraints. At last, k clusters are selected from Θ as aspects[2]. Based on the idea that the frequent clusters are usually the important aspects of customers' interests, we select the top k clusters having the highest sum of members' frequencies of occurrence. From the k aspects, the nouns and noun phrases (e.g., "speed", "size") are recognized as explicit features, and the adjectives and verbs (e.g., "fast", "big"), are recognized as implicit features.

3.2 Domain-specific Similarity

The similarity measure aims to identify terms referring to the same aspect of a product. Prior studies (Zhai et al., 2010; Zhai et al., 2011) have shown that general semantic similarities learned from thesaurus dictionaries (e.g., WordNet) do not perform well in grouping features, mainly because the similarities between words/phrases are domain dependent. For example, "ice cream sandwich" and "operating system" are not relevant in general, but they refer to the same aspect in *cell phone* reviews[3]; "smooth" and "speed" are more similar in *cell phone* domain than they are in *hair dryer* domain. Methods based on distributional information in a domain-specific corpus are usually used to determine the domain-dependent similarities between words/phrases. However, relying completely on the corpus may not be sufficient either. For example, people usually use "inexpensive" or "great price" instead of "inexpensive price"; similarly, they use "running fast" or "great speed" instead of "fast speed". Though "inexpensive" and "price" or "fast" and "speed" refer to the same aspect, we may not find they are similar based on their context or co-occurrences in the corpus.

We propose to estimate the domain-specific similarities between candidates by incorporating both general semantic similarity and corpus-based statistical association. Formally, let G be a $n \times n$ similarity matrix, where G_{ij} is the *general semantic similarity* between candidates x_i and x_j, $G_{ij} \in [0, 1]$, $G_{ij} = 1$ when $i = j$, and $G_{ij} = G_{ji}$. We use UMBC Semantic Similarity Service[4] to get G. It combines both WordNet knowledge and statistics

[2] If k is larger than the number of clusters obtained, all the clusters are selected as aspects.

[3] *Ice Cream Sandwich* is a version of the Android mobile operating system.

[4] http://swoogle.umbc.edu/SimService/index.html

from a large web corpus to compute the semantic similarity between words/phrases (Han et al., 2013).

Let T be a $n \times n$ association matrix, where T_{ij} is the pairwise *statistical association* between x_i and x_j in the domain-specific corpus, $T_{ij} \in [0, 1]$, $T_{ij} = 1$ when $i = j$, and $T_{ij} = T_{ji}$. We use normalized pointwise mutual information (NPMI) (Bouma, 2009) as the measure of association to get T, that is,

$$NPMI(x_i, x_j) = \frac{\log \frac{Nf(x_i,x_j)}{f(x_i)f(x_j)}}{-\log \frac{f(x_i,x_j)}{N}},$$

where $f(x_i)$ (or $f(x_j)$) is the number of documents where x_i (or x_j) appears, $f(x_i, x_j)$ is the number of documents where x_i and x_j co-occur in a sentence, and N is the total number of documents in the domain-specific corpus. NPMI is the normalization of pointwise mutual information (PMI), which has the pleasant property $NPMI(x_i, x_j) \in [-1, 1]$ (Bouma, 2009). The values of NPMI are rescaled to the range of $[0, 1]$, because we want $T_{ij} \in [0, 1]$.

A candidate x_i can be represented by the i-th row in G or T, i.e., the row vector $g_{i:}$ or $t_{i:}$, which tells us what x_i is about in terms of its general semantic similarities or statistical associations to other terms. The cosine similarity between two vectors $\vec{u}$ and $\vec{v}$ can be calculated as:

$$cosine(\vec{u}, \vec{v}) = \frac{\vec{u} \cdot \vec{v}}{\| \vec{u} \| \| \vec{v} \|} = \frac{\sum_{i=1}^{n} u_i v_i}{\sqrt{\sum_{i=1}^{n} u_i^2} \sqrt{\sum_{i=1}^{n} v_i^2}}.$$

By calculating the cosine similarity between two vectors of x_i and x_j ($i \neq j$), we get the following similarity metrics:

$$sim_g(x_i, x_j) = cosine(g_{i:}, g_{j:}),$$
$$sim_t(x_i, x_j) = cosine(t_{i:}, t_{j:}),$$
$$sim_{gt}(x_i, x_j) = max(cosine(g_{i:}, t_{j:}), cosine(t_{i:}, g_{j:})).$$

$sim_g(x_i, x_j)$ provides the comparison between $g_{i:}$ and $g_{j:}$. Similar row vectors in G indicate similar semantic meanings of two terms (e.g., "price" and "inexpensive"). $sim_t(x_i, x_j)$ provides the comparison between $t_{i:}$ and $t_{j:}$. Similar row vectors in T indicate similar context of two terms in the domain, and terms that occur in the same contexts tend to have similar meanings (Harris, 1954) (e.g., "ice cream sandwich" and "operating system"). $sim_{gt}(x_i, x_j)$ provides the comparison between the row vector in G and the row vector in T of two terms. $sim_{gt}(x_i, x_j)$ is designed to get high value when the terms strongly associated with x_i (or x_j) are semantically similar to x_j (or x_i). By this measure, the domain-dependent synonyms such as "smooth" and "speed" (in *cell phone* domain) can be identified because the word "smooth" frequently co-occurs with some other words (e.g., "fast", "run") that are synonymous with the word "speed".

Because $G_{ij} \in [0, 1]$ and $T_{ij} \in [0, 1]$, the values of $sim_g(x_i, x_j)$, $sim_t(x_i, x_j)$, and $sim_{gt}(x_i, x_j)$ range from 0 to 1. In addition, $sim_g(x_i, x_j) = sim_g(x_j, x_i)$, $sim_t(x_i, x_j) = sim_t(x_j, x_i)$ and $sim_{gt}(x_i, x_j) = sim_{gt}(x_j, x_i)$. When $i = j$, we set all the similarity metrics between x_i and x_j to 1. Finally, the domain-specific similarity between x_i and x_j ($i \neq j$) is defined as the weighted sum of the above three similarity metrics: $sim(x_i, x_j) = w_g sim_g(x_i, x_j) + w_t sim_t(x_i, x_j) + w_{gt} sim_{gt}(x_i, x_j)$, where w_g, w_t and w_{gt} denote the relative weight of importance of the three similarity metrics, respectively. The values of the weight ranges from 0 to 1, and $w_g + w_t + w_{gt} = 1$.

Based on the domain-specific similarities between candidates, we now define the distance measures of clustering as:

$$dist_{avg}(C_l, C_m) = \frac{\sum_{x_{i'} \in C_l} \sum_{x_{j'} \in C_m} (1 - sim(x_{i'}, x_{j'}))}{|C_l| \times |C_m|},$$
$$r(C_l) = argmax_{x_{i'} \in C_l} f(x_{i'}),$$
$$dist_{rep}(C_l, C_m) = 1 - sim(r(C_l), r(C_m)),$$

where $dist_{avg}(C_l, C_m)$ is the average of candidate distances between clusters C_l and C_m, $r(C_l)$ is the most frequent member (i.e., *representative term*) in cluster C_l, and $dist_{rep}(C_l, C_m)$ is the distance between the representative terms of two clusters. The two clusters describing the same aspect should be close to each other in terms of both average distance and representative distance, thus the final distance is defined as the maximum of these two:

$$dist(C_l, C_m) = max(dist_{avg}(C_l, C_m), dist_{rep}(C_l, C_m)).$$

3.3 Merging Constraints

Prior studies (Wagstaff et al., 2001) have explored the idea of incorporating background knowledge as constraints on the clustering process to further improve the performance. Two types of constraints are usually considered: must-link constraints specifying

that two objects (e.g., words) must be placed in the same cluster, and cannot-link constraints specifying that two objects cannot be placed in the same cluster. We also add problem-specific constraints that specify which clusters cannot be merged together, but instead of manually creating the cannot-links between specific words, our cannot-link constraints are automatically calculated during the clustering process.

Specifically, two clusters cannot be merged if they violate any of the three merging constraints: (1) The distance between two clusters must be less than a given value δ (see Algorithm 1). (2) There must be at least one noun or noun phrase (candidate of explicit feature) existing in one of the two clusters. Because we assume an aspect should contain at least one explicit feature, and we would not get an aspect by merging two non-aspect clusters. (3) The sum of frequencies of the candidates from two clusters co-occurring in the same sentences must be higher than the sum of frequencies of them co-occurring in the same documents but different sentences. The idea is that people tend to talk about different aspects of a product in different sentences in a review, and talk about the same aspect in a small window (e.g., the same sentence).

4 Experiments

In this section, we evaluate the effectiveness of the proposed approach on feature extraction and aspect discovery. Table 1 describes the datasets from three different domains that were used in the experiments. The cell phone reviews were collected from the online shop of a cell phone company, and the GPS and TV reviews were collected from Amazon.

Three human annotators manually annotate the datasets to create gold standards of features and aspects. These annotators first identify feature expressions from reviews independently. The expressions that were agreed by at least two annotators were selected as features. Then the authors manually specified a set of aspects based on these features, and asked three annotators to label each feature with an aspect category. The average inter-annotator agreement on aspect annotation was $\kappa = 0.687$ ($stddev = 0.154$) according to Cohen's Kappa statistic. To obtain the gold standard annotation of aspects, the annotators discussed to reach an agreement when there was a disagreement on the aspect category of a feature. *We are making the datasets and annotations publicly available*[5].

Table 1 shows the number of reviews, aspects, unique explicit/implicit features manually identified by annotators, and candidates of explicit (i.e., noun and noun phrase) and implicit (i.e., adjective and verb) features extracted from the datasets in three domains.

	Cell phone	GPS	TV
# Reviews	500	500	500
# Aspects	46	37	34
# Features (expl.)	419	637	485
# Features (impl.)	339	492	277
# Candidates (expl.)	1,248	2,078	2,333
# Candidates (impl.)	1,115	1,779	1,690

Table 1: Data sets and gold standards.

We use "CAFE" (**C**lustering for **A**spect and **F**eature **E**xtraction) to denote the proposed method. We assume the number of aspects k is specified by the users, and set $k = 50$ throughout all the experiments. We use $s = 500$, $\delta = 0.8$, $w_g = w_t = 0.2, w_{gt} = 0.6$ as the default setting of CAFE, and study the effect of parameters in Section "Influence of Parameters". In addition, we evaluate each individual similarity metric – "CAFE-g", "CAFE-t" and "CAFE-gt" denote the variations of "CAFE" that use $sim_g(x_i, x_j)$, $sim_t(x_i, x_j)$, and $sim_{gt}(x_i, x_j)$ as the similarity measure, respectively. We empirically set $\delta = 0.4$ for "CAFE-g", $\delta = 0.84$ for "CAFE-t" and $\delta = 0.88$ for "CAFE-gt".

4.1 Evaluations on Feature Extraction

We compared CAFE against the following two state-of-the-art methods on feature extraction:

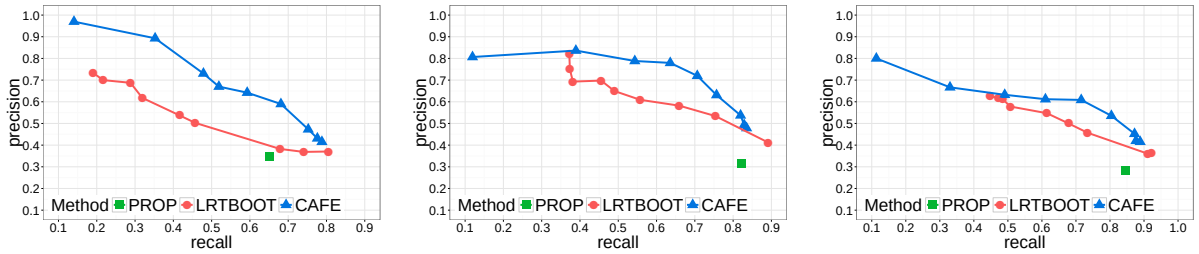
- **PROP**: A double propagation approach (Qiu et al., 2009) that extracts features using handcrafted rules based on dependency relations between features and opinion words.
- **LRTBOOT**: A bootstrapping approach (Hai et al., 2012) that extracts features by mining pairwise feature-feature, feature-opinion, opinion-opinion associations between terms in the corpus, where the association is measured by the likelihood ratio tests (LRT).

Both methods require seeds terms. We ranked the feature candidates by descending document fre-

[5] http://knoesis.wright.edu/researchers/luchen/download/naacl16_aspects.zip

	Cell-phone			GPS			TV			
Method	Precision	Recall	F-score	Precision	Recall	F-score	Precision	Recall	F-score	macro-averaged F-score
PROP	0.3489	0.6503	0.4541	0.3157	**0.8222**	0.4562	0.2851	**0.8454**	0.4264	0.4456
LRTBOOT	0.3819	**0.8112**	0.5193	0.5342	0.7488	0.6235	0.4572	0.7340	0.5635	0.5688
CAFE	0.6421	0.5929	0.6165	**0.7197**	0.7064	**0.7130**	**0.6086**	0.7155	**0.6577**	**0.6624**
CAFE-g	**0.6822**	0.5667	**0.6191**	0.6831	0.6154	0.6475	0.5959	0.6330	0.6139	0.6268
CAFE-t	0.4761	0.5833	0.5243	0.5765	0.6845	0.6259	0.4892	0.7175	0.5817	0.5773
CAFE-gt	0.5519	0.6000	0.5749	0.6512	0.6028	0.6261	0.5445	0.7320	0.6245	0.6085

Table 2: Experimental results of feature extraction.



(a) Cell Phone (b) GPS (c) TV

Figure 1: Precision-recall curves at various parameter settings for three feature extraction methods.

quency and manually selected the top 10 genuine features as seeds for them. According to the study (Hai et al., 2012), the performance for LRTBOOT remained almost constant when increasing the seeds from 1 to 50. Three association thresholds need to be specified for LRTBOOT. Following the original study in which the experiments were conducted on cell-phone reviews, we set $ffth = 21.0$, $ooth = 12.0$, and performed grid search for the value of $foth$. The best results were achieved at $foth = 9.0$ for cell-phone reviews, and at $foth = 12.0$ for GPS and TV reviews.

The results were evaluated by precision $= \frac{N_{agree}}{N_{result}}$, recall $= \frac{N_{agree}}{N_{gold}}$, and F-score $= \frac{2\times\text{precision}\times\text{recall}}{\text{precision}+\text{recall}}$, where N_{result} and N_{gold} are the number of features in the result and the gold standard, respectively, and N_{agree} is the number of features that are agreed by both sides. Because PROP and LRTBOOT extract only explicit features, the evaluation was conducted on the quality of explicit features. The performance of identifying implicit features will be examined by evaluation on aspect discovery, because implicit features have to be merged into aspects to be detected.

Table 2 shows the best results (in terms of F-score) of feature extraction by different methods. Both PROP and LRTBOOT obtain high recall and relatively low precision. CAFE greatly improves precision, with a relatively small loss of recall, resulting in 21.68% and 9.36% improvement in macro-averaged F-score over PROP and LRTBOOT, respectively. We also plot precision-recall curves at various parameter settings for CAFE and LRTBOOT in Figure 1. For CAFE, we kept $s = 500$, $w_g = w_t = 0.2, w_{gt} = 0.6$, and increased δ from 0.64 to 0.96. For LRTBOOT, we kept $ffth = 21.0$, $ooth = 12.0$, and increased $foth$ from 6.0 to 30.0. For PROP, only one precision-recall point was obtained. From Figure 1, we see that the curve of CAFE lies well above those of LRTBOOT and PROP across three datasets. Though LRTBOOT achieved similar precision as CAFE did at the recall rate of approximately 0.37 for GPS reviews and at the recall rate of approximately 0.49 for TV reviews, it performed worse than CAFE at increasing recall levels for both datasets.

The key difference between CAFE and the baselines is that CAFE groups terms into clusters and identifies the terms in the selected aspect clusters as features, while both baselines enlarge a feature seed set by mining syntactical or statistical associations between features and opinion words. The results suggest that features can be more precisely identified via aspect clustering. Generally, CAFE is superior to its variations, and CAFE-g outperforms CAFE-gt and CAFE-t.

4.2 Evaluations on Aspect Discovery

For comparison with CAFE on aspect discovery, we implemented the following three methods:

- **MuReinf**: A clustering method (Su et al., 2008) that utilizes the mutual reinforcement as-

sociation between features and opinion words to iteratively group them into clusters. Similar to the proposed method, it is unsupervised, clustering-based, and handling implicit features.

- **L-EM**: A semi-supervised learning method (Zhai et al., 2011) that adapts the Naive Bayesian-based EM algorithm to group synonym features into categories. Because semi-supervised learning needs some labeled examples, the proposed method first automatically generates some labeled examples (i.e., the groups of synonym feature expressions) based on features sharing common words and lexical similarity.
- **L-LDA**: A baseline method (Zhai et al., 2011) that is based on LDA. The same labeled examples generated by L-EM are used as seeds for each topic in topic modeling.

These three methods require features to be extracted beforehand, and focus on grouping features into aspects. Both LRTBOOT and CAFE are used to provide the input features to them. We set $\alpha = 0.6$ for MuReinf, because their study (Su et al., 2008) showed that the method achieved best results at $\alpha > 0.5$. All three methods utilize dictionary-based semantic similarity to some extent. Since CAFE uses the UMBC Semantic Similarity Service, we use the same service to provide the semantic similarity for all the methods.

	Cell-phone	GPS	TV	macro-average
LRTBOOT + MuReinf	0.7182	0.8031	0.7747	0.7653
LRTBOOT + L-EM	0.6633	0.6893	0.7138	0.6888
LRTBOOT + L-LDA	0.7653	0.7198	0.7664	0.7505
CAFE + MuReinf	0.7973	0.8212	**0.8334**	0.8173
CAFE + L-EM	0.7581	0.7772	0.7879	0.7744
CAFE + L-LDA	0.7904	0.8144	0.8247	0.8098
CAFE	0.8041	**0.8238**	0.8326	**0.8202**
CAFE-g	0.7382	0.7534	0.8205	0.7707
CAFE-t	0.7868	0.8050	0.7965	0.7961
CAFE-gt	**0.8073**	0.7716	0.7906	0.7898

Table 3: Rand Index of aspect identification.

The results were evaluated using Rand Index (Rand, 1971), a standard measure of the similarity between the clustering results and a gold standard. Given a set of n objects and two partitions of them, the Rand Index is defined as $\frac{2(a+b)}{n\times(n-1)}$. The idea is that the agreements/disagreements between two partitions are checked on $n \times (n-1)$ pairs of objects. Among all the pairs, there are a pairs belonging to the same cluster in both partitions, and b pairs belonging to different clusters in both partitions. In this study, the gold standard and the aspect clusters may not share the exact same set of features due to the noise in feature extraction, therefore we consider n the number of expressions in the union of two sets.

Table 3 shows the Rand Index achieved by different methods. Among the methods that generate partitions of the same features provided by CAFE, CAFE achieves the best macro-averaged Rand Index, followed by CAFE + MuReinf, CAFE + L-LDA, and CAFE + L-EM. CAFE outperforms the variations using the single similarity metric, i.e., CAFE-g, CAFE-t and CAFE-gt. The results imply the effectiveness of our domain-specific similarity measure in identifying synonym features in a particular domain. Using the input features from LRTBOOT, the performance of MuReinf, L-EM and L-LDA decrease on all three domains, compared with using the input features from CAFE. The decrease is more significant for L-EM and L-LDA than for MuReinf, which suggest that the semi-supervised methods L-EM and L-LDA are more dependent on the quality of input features.

Table 4 illustrates a sample of the discovered aspects and features by CAFE. The algorithm identifies the important *aspects* in general sense as well as the important aspects that are not so obvious thus could be easily missed by human judges, e.g., *suction cup* for GPS and *glare* for TV. In addition, both explicit and implicit features are identified and grouped into the aspects, e.g., *expensive* and *price*, *big* and *size*, *sensitive* and *signal*, etc.

4.3 Influence of Parameters

We varied the value of δ (distance *upper bound*), s (the number of frequent candidates selected to generate seed clusters) and w_{gt} (the weight of sim_{gt}) to see how they impact the results of CAFE, for both feature extraction (in terms of F-Score) and aspect discovery (in terms of Rand Index). Both F-score and Rand Index increases rapidly at first and then slowly decreases as we increase δ from 0.64 to 0.96 (see the left subplot in Figure 2). Because more clusters are allowed to be merged as we increase δ, which is good at first but then it introduces more noise than benefit. Based on the experiments on

Cell-phone	GPS	TV
screen, **display**, **touch**, **button**, **pixel screen**, **icon**, **amold display**, *pressed*, *click*, *navigate*	**direction**, **route**, **road**, **instructions**, **streets**, **highway**, **side**, **lane**, **exit**, **intersection**, **track**	**picture**, **hd picture**, **image**, **scene**, **photo**, **action scenes**, **view**, *visual*, *show*, *present*
battery, **life**, **battery life**, **power**, **backup battery**, *spare*, *recharge*, *powered*, *plug*, *lasted*	**map**, **point**, **information**, **interest**, **info**, **data**, **map loading**, *accurate*, *search*, *locate*, *listed*	**cable**, **channel**, **cable box**, **station**, **wire**, **antenna tuner**, **format**, **transmission**
camera, **picture**, **video**, **photo**, **zoom**, **motion videos**, **gallery**, *fuzzy*, *grainy*, *shooting*, *recorded*	**signal**, **satellite**, **antenna**, **receiver**, **radio**, **fm transmitter**,**traffic receiver**, **sensor**, *sensitive*	**sound**, **speaker**, **volume**, **noise**, **hum**, **echo**, *audible*, *tinny*, *muffled*, *hissing*, *loud*, *pitched*
call, **car**, **speaker**, **call quality**, **call reminder**, *drop*, *connect*, *answered*, *clear*, *hear*, *speak*	**voice**, **voice recognition**, **microphone**, **speaker**, **volume**, **guy voice**, **robot voice**, *repeat*, *loud*	**price**, **market**, **cost**, **tax**, **credit**, **sale**, **discount**, *purchase*, *expensive*, *worth*, *saved*, *cheap*
size, **hand**, **screen size**, **finger**, **font size**, **width**, *tiny*, *huge*, *bigger*, *larger*, *big*, *carry*, *small*, *large*	**suction cup**, **windshield**, **bean bag**, **mount**, **attachment**, *unit fall*, *attaching*, *pulling*, *break*	**glare**, **reflection**, **sunlight**, **lamp**, **daylight**, *blind*, *flickering*, *dim*, *fluorescent*, *dark*, *reflective*

Table 4: Examples of discovered aspects and features by the proposed approach CAFE. Explicit and implicit features are denoted in boldface and italics, respectively. The first term in each cluster is the representative term of that aspect.

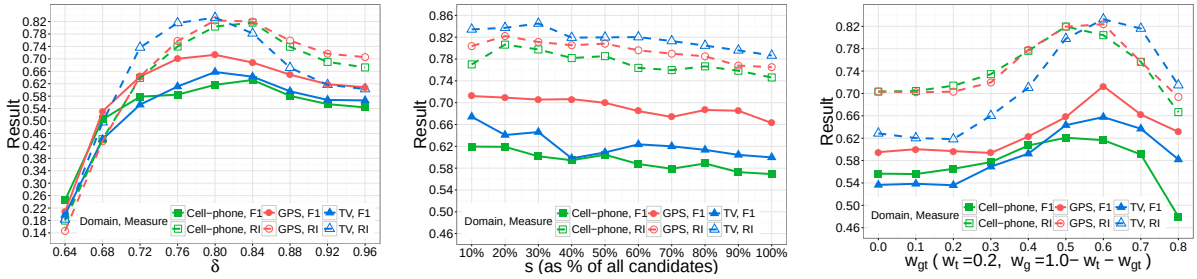


Figure 2: CAFE parameter tuning: feature quality in terms of F-score (F1) and aspect quality in terms of Rand Index (RI). The default setting is $s = 500$, $\delta = 0.8$, $w_g = w_t = 0.2$, $w_{gt} = 0.6$. We keep other parameters as the default setting when we tune an individual parameter.

three domains, the best results can be achieved when δ is set to a value between 0.76 and 0.84. The middle subplot illustrates the impact of s, which shows that CAFE generates better results by first clustering the top 10%-30% most frequent candidates. Infrequent words/phrases are usually more noisy, and the results could be affected more seriously if the noises are included in the clusters in the early stage of clustering. Experiments were also conducted to study the impact of the three similarity metrics. Due to the space limit, we only display the impact of w_{gt} and w_g given $w_t = 0.2$. As we can see from the right subplot in Figure 2, setting w_{gt} or w_g to zero evidently decreases the performance, indicating both similarity metrics are useful. The best F-score and Rand Index can be achieved when we set w_{gt} to 0.5 or 0.6 across all three domains.

5 Conclusion

In this paper, we proposed a clustering approach that simultaneously extracts features and aspects of a given product from reviews. Our approach groups the feature candidates into clusters based on their domain-specific similarities and merging constraints, then selects the important aspects and identifies features from these aspects. This approach has the following advantages: (1) It identifies both aspects and features simultaneously. The evaluation shows its accuracy on both tasks outperforms the competitors. (2) Both explicit and implicit features can be identified and grouped into aspects. The mappings of implicit features into explicit features are accomplished naturally during the clustering process. (3) It does not require labeled data or seed words, which makes it easier to apply and broader in application. In our future work, instead of selecting aspects based on frequency, we will leverage domain knowledge to improve the selection.

Acknowledgments

We would like to thank Lushan Han and UMBC Ebiquity Lab for kindly allowing us to access their Semantic Similarity Service. This research was partially supported by NSF awards CNS-1513721 “Context-Aware Harassment Detection on Social Media” and EAR-1520870 “Hazards SEES: Social and Physical Sensing Enabled Decision Support for Disaster Management and Response”.

References

Wouter Bancken, Daniele Alfarone, and Jesse Davis. 2014. Automatically detecting and rating product aspects from textual customer reviews. In *DMNLP Workshop at ECML/PKDD*.

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022.

Gerlof Bouma. 2009. Normalized (pointwise) mutual information in collocation extraction. *GSCL*, pages 31–40.

Samuel Brody and Noemie Elhadad. 2010. An unsupervised aspect-sentiment model for online reviews. In *NAACL*, pages 804–812.

Giuseppe Carenini, Raymond T Ng, and Ed Zwart. 2005. Extracting knowledge from evaluative text. In *K-CAP*, pages 11–18.

Zhiyuan Chen, Arjun Mukherjee, Bing Liu, Meichun Hsu, Malu Castellanos, and Riddhiman Ghosh. 2013. Exploiting domain knowledge in aspect extraction. In *EMNLP*, pages 1655–1667.

Zhiyuan Chen, Arjun Mukherjee, and Bing Liu. 2014. Aspect extraction with automated prior knowledge learning. In *ACL*, pages 347–358.

Geli Fei, Bing Liu, Meichun Hsu, Malu Castellanos, and Riddhiman Ghosh. 2012. A dictionary-based approach to identifying aspects im-plied by adjectives for opinion mining. In *COLING*, page 309.

Honglei Guo, Huijia Zhu, Zhili Guo, XiaoXun Zhang, and Zhong Su. 2009. Product feature categorization with multilevel latent semantic association. In *CIKM*, pages 1087–1096.

Zhen Hai, Kuiyu Chang, and Jung-jae Kim. 2011. Implicit feature identification via co-occurrence association rule mining. In *Computational Linguistics and Intelligent Text Processing*, pages 393–404.

Zhen Hai, Kuiyu Chang, and Gao Cong. 2012. One seed to find them all: mining opinion features via association. In *CIKM*, pages 255–264.

Lushan Han, Abhay Kashyap, Tim Finin, James Mayfield, and Jonathan Weese. 2013. Umbc ebiquity-core: Semantic textual similarity systems. In *the Second Joint Conference on Lexical and Computational Semantics*, pages 44–52.

Zellig S Harris. 1954. Distributional structure. *Word*.

Thomas Hofmann. 2001. Unsupervised learning by probabilistic latent semantic analysis. *Machine learning*, 42(1-2):177–196.

Minqing Hu and Bing Liu. 2004. Mining opinion features in customer reviews. In *AAAI*, pages 755–760.

Wei Jin, Hung Hay Ho, and Rohini K Srihari. 2009. Opinionminer: a novel machine learning system for web opinion mining and extraction. In *SIGKDD*, pages 1195–1204.

Yohan Jo and Alice H Oh. 2011. Aspect and sentiment unification model for online review analysis. In *WSDM*, pages 815–824.

Suin Kim, Jianwen Zhang, Zheng Chen, Alice H Oh, and Shixia Liu. 2013. A hierarchical aspect-sentiment model for online reviews. In *AAAI*.

Chenghua Lin and Yulan He. 2009. Joint sentiment/topic model for sentiment analysis. In *CIKM*, pages 375–384.

Bing Liu, Minqing Hu, and Junsheng Cheng. 2005. Opinion observer: analyzing and comparing opinions on the web. In *WWW*, pages 342–351.

Yue Lu, ChengXiang Zhai, and Neel Sundaresan. 2009. Rated aspect summarization of short comments. In *WWW*, pages 131–140.

Samaneh Moghaddam and Martin Ester. 2011. Ilda: interdependent lda model for learning latent aspects and their ratings from online product reviews. In *SIGIR*, pages 665–674.

Arjun Mukherjee and Bing Liu. 2012. Aspect extraction through semi-supervised modeling. In *ACL*, pages 339–348.

Guang Qiu, Bing Liu, Jiajun Bu, and Chun Chen. 2009. Expanding domain sentiment lexicon through double propagation. In *IJCAI*, pages 1199–1204.

William M Rand. 1971. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical association*, 66(336):846–850.

Qi Su, Kun Xiang, Houfeng Wang, Bin Sun, and Shiwen Yu. 2006. Using pointwise mutual information to identify implicit features in customer reviews. In *ICCPOL*, pages 22–30.

Qi Su, Xinying Xu, Honglei Guo, Zhili Guo, Xian Wu, Xiaoxun Zhang, Bin Swen, and Zhong Su. 2008. Hidden sentiment association in chinese web opinion mining. In *WWW*, pages 959–968.

Ivan Titov and Ryan McDonald. 2008a. Modeling online reviews with multi-grain topic models. In *WWW*, pages 111–120.

Ivan Titov and Ryan T McDonald. 2008b. A joint model of text and aspect ratings for sentiment summarization. In *ACL*, pages 308–316.

Kiri Wagstaff, Claire Cardie, Seth Rogers, Stefan Schrödl, et al. 2001. Constrained k-means clustering with background knowledge. In *ICML*, volume 1, pages 577–584.

Hao Wang and Martin Ester. 2014. A sentiment-aligned topic model for product aspect rating prediction. In *EMNLP*, pages 1192–1202.

Liheng Xu, Kang Liu, Siwei Lai, Yubo Chen, and Jun Zhao. 2013. Mining opinion words and opinion targets in a two-stage framework. In *ACL*, pages 1764–1773.

Yinqing Xu, Tianyi Lin, Wai Lam, Zirui Zhou, Hong Cheng, and Anthony Man-Cho So. 2014. Latent aspect mining via exploring sparsity and intrinsic information. In *CIKM*, pages 879–888.

Bishan Yang and Claire Cardie. 2013. Joint inference for fine-grained opinion extraction. In *ACL*, pages 1640–1649.

Jianxing Yu, Zheng-Jun Zha, Meng Wang, and Tat-Seng Chua. 2011. Aspect ranking: identifying important product aspects from online consumer reviews. In *ACL*, pages 1496–1505.

Zhongwu Zhai, Bing Liu, Hua Xu, and Peifa Jia. 2010. Grouping product features using semi-supervised learning with soft-constraints. In *COLING*, pages 1272–1280.

Zhongwu Zhai, Bing Liu, Hua Xu, and Peifa Jia. 2011. Clustering product features for opinion mining. In *WSDM*, pages 347–354.

Lei Zhang and Bing Liu. 2014. Aspect and entity extraction for opinion mining. In *Data Mining and Knowledge Discovery for Big Data*, pages 1–40.

Yu Zhang and Weixiang Zhu. 2013. Extracting implicit features in online customer reviews for opinion mining. In *WWW companion*, pages 103–104.

Li Zhao, Minlie Huang, Haiqiang Chen, Junjun Cheng, and Xiaoyan Zhu. 2014. Clustering aspect-related phrases by leveraging sentiment distribution consistency. In *EMNLP*, pages 1614–1623.