Improving Opinion Aspect Extraction Using Semantic Similarity and Aspect Associations

Qian Liu[1,2], **Bing Liu**[3], **Yuanlin Zhang**[4], **Doo Soon Kim**[5] and **Zhiqiang Gao**[1,2]
[1]Key Lab of Computer Network and Information Integration (Southeast University), Ministry of Education, China
[2]School of Computer Science and Engineering, Southeast University, China
[3]Department of Computer Science, University of Illinois at Chicago, USA
[4]Department of Computer Science, Texas Tech University, USA
[5]Bosch research lab, USA
[1,2]{qianliu, zqgao}@seu.edu.cn, [3]liub@cs.uic.edu, [4]y.zhang@ttu.edu, [5]DooSoon.Kim@us.bosch.com

Abstract

Aspect extraction is a key task of fine-grained opinion mining. Although it has been studied by many researchers, it remains to be highly challenging. This paper proposes a novel unsupervised approach to make a major improvement. The approach is based on the framework of lifelong learning and is implemented with two forms of recommendations that are based on semantic similarity and aspect associations respectively. Experimental results using eight review datasets show the effectiveness of the proposed approach.

Introduction

Aspect extraction is a fundamental task of opinion mining or sentiment analysis (Liu 2012). It aims to extract opinion targets from opinion text. For example, from "*This phone has a good screen*," it aims to extract "screen." In product reviews, aspects are product attributes or features. They are needed in many sentiment analysis applications.

Aspect extraction has been studied by many researchers. There are two main approaches: *supervised* and *unsupervised*. Some existing work has shown that unsupervised syntactic dependency-based methods such as *double propagation* (DP) (Qiu et al. 2011) can perform better than supervised Conditional Random Fields (CRF) (Lafferty, McCallum, and Pereira 2001) based methods. The unsupervised dependency-based methods also have the key advantage of not requiring any human labeled data. They are based on the fact that opinions have targets and there are often explicit syntactic relations between opinion words (e.g., "good") and target aspects (e.g., "screen"). By exploiting such relations, DP and other related methods can use a set of seed opinion words to extract aspects and new opinion words, and then use them to extract more aspects and opinion words through bootstrapping propagation.

Figure 1 shows the dependency relations between words in "*The phone has a good screen*." If "good" is a known opinion word (given or extracted), "screen," a noun modified by "good," is clearly an aspect as they have a dependency relation *amod*. From a given set of opinion words, we can extract a set of aspects if we have a syntactic rule like "*if a word A, whose part-of-speech (POS) is a singular noun (nn), has the dependency relation amod with (i.e., modified by) an opinion word O, then A is an aspect*." Similarly, one can use such rules to extract new aspects from the extracted aspects, and new opinion words from the extracted aspects.

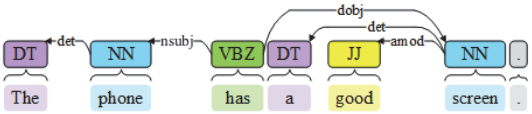

Figure 1: Dependency relations in the sentence "*The phone has a good screen*."

Although effective, syntactic rule-based methods such as DP still have room for major improvements. This is not surprising as it is very hard to design a set of rules to perform extraction with high precision and high recall due to the flexibility of natural languages. One way to improve is to use the prior knowledge in the framework of *lifelong machine learning*. That is, the system retains its past experiences and learned results as knowledge and uses it to help the new learning or extraction. In other words, if the system already knows a lot before extraction, it clearly can do much better. The prior knowledge is mined automatically by exploiting the abundance of reviews for all kinds of products on the Web. This idea is workable because many products actually share aspects, e.g., many electronic products have aspects screen and battery. To exploit such knowledge for new extraction, we use the idea of recommendation, in particular collaborative filtering (Adomavicius and Tuzhilin 2005). This type of recommendation uses the behavioral information of other users to recommend products/services to the current user. This is exactly the idea that we want to employ, using the information in reviews of a large number of other products to help extract aspects from reviews of the current product. To the best of our knowledge, this is the first time that recommendation is used for aspect extraction.

In this work, we propose to use DP as the base and improve its results dramatically through aspect recommendation. Two forms of recommendations are proposed, (1) semantic similarity-based, and (2) aspect associations-based. Semantic similarity-based recommendation aims to solve

the problem of missing synonymous aspects of DP using word vectors trained from a large corpus of 5.8 million reviews for similarity comparison. The word vectors are regarded as a form of prior knowledge learned from the past data. For example, "photo" is a synonym of "picture." Using the DP method, "picture" is extracted as an aspect from the sentence "*The picture is blurry*," but "photo" is not extracted from the sentence "*The phone is good, but not its photos*." One reason for the inability to extract "photo" is that to ensure good extraction precision and recall, many useful rules with low precision are not used. The proposed semantic similarity-based recommendation makes use of the extracted aspect "picture" to recommend "photo" based on the semantic similarity of the two words.

However, "picture" cannot be used to recommend "battery" as an aspect because their semantic similarity value is very small. To recommend "battery" (if it is not extracted), we use the second form of recommendation, i.e., aspect associations or correlations. The idea is that many aspects are correlated or co-occur across domains. For example, those products with the aspect "picture" also have a high chance of using batteries as pictures are usually taken by digital devices which need batteries. If such associations can be discovered, they can be used in recommendation of additional aspects. For this purpose, we employ association rules from data mining (Agrawal and Srikant 1994) which fit our needs very well. To mine associations, we use the extraction results from reviews of many other products or domains in the lifelong learning fashion (Chen and Liu 2014).

In our experiments, we use a popular aspect extraction evaluation corpus from (Hu and Liu 2004) and a new corpus from (Liu et al. 2015). To learn word vectors and aspect associations, we use two large collections of product reviews. Experimental results show that the two forms of recommendations can recommend very reliable aspects, and the approach that employs both recommendations outperforms the state-of-the-art dependency rule-based methods markedly.

Related Work

There are two main approaches to aspect extraction: *supervised* and *unsupervised*. The former is mainly based on CRF (Jakob and Gurevych 2010; Choi and Cardie 2010; Mitchell et al. 2013), while the latter is mainly based on topic modeling (Mei et al. 2007; Titov and McDonald 2008; Li, Huang, and Zhu 2010; Brody and Elhadad 2010; Wang, Lu, and Zhai 2010; Moghaddam and Ester 2011; Mukherjee and Liu 2012), and syntactic rules designed using dependency relations (Zhuang, Jing, and Zhu 2006; Wang and Wang 2008; Wu et al. 2009; Zhang et al. 2010; Qiu et al. 2011).

On the supervised approach, CRF-based methods need manually labeled training data. Our method is unsupervised. On the unsupervised approach, topic modeling often only gives some rough topics rather than precise aspects as a topical term does not necessarily mean an aspect. For example, in a battery topic, a topic model may find topical terms such as "battery," "life," and "time," etc., which are related to battery life (Lin and He 2009; Zhao et al. 2010; Jo and Oh 2011; Fang and Huang 2012), but each word is not an aspect.

There are also frequency-based methods (Hu and Liu 2004; Popescu and Etzioni 2005; Zhu et al. 2009), word alignment methods (Liu et al. 2013), label propagation methods (Zhou, Wan, and Xiao 2013), and other methods.

This paper is most related to the DP method (Qiu et al. 2011), and aims to improve it. Since our method employs word vectors learned from a large collection of product reviews, it is also related to (Xu, Liu, and Zhao 2014), which proposed a joint opinion relation detection method OCDNN. Although they also used word vectors to represent words in the neural network training, they used that as a feature representation in their classification. The work (Pavlopoulos and Androutsopoulos 2014) explored the word vectors trained on English Wikipedia to compute word similarities used in a clustering algorithm. However, our work is quite different, we train word vectors using a large review corpus and use them to recommend aspects.

Our work is also related to topic modeling-based methods in (Chen and Liu 2014; Chen, Mukherjee, and Liu 2014) as they also used multiple past domains to help aspect extraction in the lifelong learning fashion. However, they can only find some rough topics as other topic models. We can find more precise aspects with the help of multiple past domains. In (Liu et al. 2015), a rule selection method is proposed to improve DP, but it is a supervised method.

Overall Algorithm

This section introduces algorithm AER (Aspect Extraction based on Recommendation), Algorithm 1, which consists of two main steps: base extraction and recommendation.

Algorithm 1 AER($\mathcal{D}^t$, $\mathcal{R}^-$, $\mathcal{R}^+$, $\mathcal{O}$)

Input: Target dataset $\mathcal{D}^t$, high precision aspect extraction rules $\mathcal{R}^-$, high recall aspect extraction rules $\mathcal{R}^+$, seed opinion words $\mathcal{O}$
Output: Extracted aspect set $\mathcal{A}$

1: $\mathcal{T}^- \leftarrow$ DPextract($\mathcal{D}^t$, $\mathcal{R}^-$, $\mathcal{O}$);
2: $\mathcal{T}^+ \leftarrow$ DPextract($\mathcal{D}^t$, $\mathcal{R}^+$, $\mathcal{O}$);
3: $\mathcal{T} \leftarrow \mathcal{T}^+ - \mathcal{T}^-$;
4: $\mathcal{T}^s \leftarrow$ Sim-recom($\mathcal{T}^-$, $\mathcal{T}$);
5: $\mathcal{T}^a \leftarrow$ AR-recom($\mathcal{T}^-$, $\mathcal{T}$);
6: $\mathcal{A} \leftarrow \mathcal{T}^- \cup \mathcal{T}^s \cup \mathcal{T}^a$.

Step 1 (base extraction, lines 1-2): Given the target document collection $\mathcal{D}^t$ for extraction and a set $\mathcal{O}$ of seed opinion words, this step first uses the DP method (DPextract) to extract an initial (or base) set $\mathcal{T}^-$ of aspects employing a set $\mathcal{R}^-$ of high precision rules (line 1). The set of high precision rules are selected from the set of rules in DP by evaluating their precisions individually using a development set. The set $\mathcal{T}^-$ of extracted aspects thus has very high precision but not high recall. Then, extract a set $\mathcal{T}^+$ of aspects from a larger set $\mathcal{R}^+$ of high recall rules ($\mathcal{R}^- \subseteq \mathcal{R}^+$) also using DPextract (line 2). The set $\mathcal{T}^+$ of extracted aspects thus has very high recall but not high precision.

Step 2 (recommendation, lines 3-6): This step recommends more aspects using $\mathcal{T}^-$ as the base to improve the recall. To ensure recommendation quality, we require that the

aspects must be from the set $\mathcal{T} = \mathcal{T}^+ - \mathcal{T}^-$ (line 3). Two forms of recommendation are performed, similarity-based using Sim-recom (line 4) and association rule-based using AR-recom (line 5). Their respective results $\mathcal{T}^s$ and $\mathcal{T}^a$ are combined with $\mathcal{T}^-$ to produce the final result. The next two sections detail the two recommendation methods.

Similarity-based Recommendation

Similarity-based recommendation can be implemented with a few methods. Here we focus only on the word vector-based method as it performs the best compared with a few strong baselines (see the experiments for the comparison).

Word Vectors

In this work, word vectors trained using neural networks are employed to compute semantic similarities of aspect terms. Researchers have shown that using word vectors trained this way is highly effective for the purpose of semantic similarity comparison (Joseph, Lev, and Yoshua 2010; Mikolov et al. 2013). There are several publicly available word vector resources trained from Wikipedia, Reuters news or Broadcast News for general NLP tasks such as POS tagging and Named Entity Recognition (Collobert and Weston 2008; Mnih and Hinton 2009; Huang et al. 2012; Pennington, Socher, and Manning 2014). However, our initial experiments with these word vectors show that they are not accurate for our task. We thus trained our own vectors using a large review corpus.

Recommending Aspects by Sim-recom

Algorithm 2 gives the details of Sim-recom($\mathcal{T}^-$, $\mathcal{T}$), which recommends aspects based on aspect similarities. For each term t in $\mathcal{T}$, which can be a single word or a multi-word phrase, if the similarity between t and any term in $\mathcal{T}^-$ is at least ϵ (line 2), which means that t is very likely to be an aspect and should be recommended, then add t into $\mathcal{T}^s$ (line 3). The final recommended aspect set is $\mathcal{T}^s$.

Algorithm 2 Sim-recom($\mathcal{T}^-$, $\mathcal{T}$)

Input: Aspect sets $\mathcal{T}^-$ and $\mathcal{T}$
Output: Recommended aspect set $\mathcal{T}^s$
1: **for** each aspect term $t \in \mathcal{T}$ **do**
2: **if** (Sim(t, $\mathcal{T}^-$) $\geq \epsilon$) **then**
3: $\mathcal{T}^s \leftarrow \mathcal{T}^s \cup \{t\}$;
4: **end if**
5: **end for**

The function Sim(t, $\mathcal{T}^-$) in line 2 returns the maximum similarity between term t and the set of terms in $\mathcal{T}^-$, i.e.,

$$\text{Sim}(t, \mathcal{T}^-) = max\{\text{VS}(\phi_t, \phi_{t_q}): t_q \in \mathcal{T}^-\}$$

where ϕ_t is t's vector, VS(ϕ_t, ϕ_{t_q}) is $\text{VS}^w(\phi_t, \phi_{t_q})$ if t and t_q are single words, otherwise, VS(ϕ_t, ϕ_{t_q}) is $\text{VS}^p(\phi_t, \phi_{t_q})$. $\text{VS}^w(\phi_t, \phi_{t_q})$ and $\text{VS}^p(\phi_t, \phi_{t_q})$ are defined as follows.

Given two terms t and t', we obtain their vectors ϕ_t and $\phi_{t'}$ from the pre-trained word vectors, and employ the cosine similarity as their semantic similarity, i.e.,

$$\text{VS}^w(\phi_t, \phi_{t'}) = \frac{\phi_t^T \phi_{t'}}{||\phi_t|| \cdot ||\phi_{t'}||}$$

Since there are no vectors for multi-word phrases in the pre-trained vectors, we use the average cosine similarities of words in the phrases to evaluate phrase similarities, i.e.,

$$\text{VS}^p(\phi_t, \phi_{t'}) = \frac{\sum_{i=1}^{M} \sum_{j=1}^{N} \text{VS}^w(\phi_{t_i}, \phi_{t'_j})}{M \times N}$$

where M is the number of single words in t, and N is that of t'. We use average similarity for multi-word phrases because it considers the length of phrases, and sets lower similarity naturally if the lengths of two phrases are different.

Association-based Recommendation

We now detail association-based recommendation. We first introduce some basic concepts before giving the details of association rule generation and the corresponding recommendation method.

Basic Concepts

Let $\mathcal{DB}$ be a *transaction database*, $\mathcal{I}$ be the set of all distinct *items* in $\mathcal{DB}$. In our case, each item is an aspect term, each *transaction* $db_i \in \mathcal{DB}$ is a set of distinct aspect terms extracted from a set of reviews of product domain $\mathcal{D}_i$.

From $\mathcal{DB}$, an association rule mining algorithm generates a set of rules that satisfies the user-specified *minimum support* and *minimum confidence* thresholds (Agrawal and Srikant 1994). Each rule is of the following form:

$$X \rightarrow Y,$$

where X and Y are disjoint sets of items, a set of aspects in our case. X and Y are called *antecedent* and *consequent* of the rule respectively. The *support* of the rule is the number of transactions that contains both X and Y divided by the total number of transactions, and the *confidence* of the rule is the number of transactions that contains both X and Y divided by the number of transactions that contains X.

Association Rule Generation

Given a review dataset $\mathcal{D}$ consisting of n domains (types of products), a set $\mathcal{O}$ of seed opinion words, a set $\mathcal{R}$ of aspect extraction rules, which can be a subset or all of $\mathcal{R}^+$ in Algorithm 1, association rule mining outputs a set $\mathcal{R}^a$ of association rules. The process consists of two phases.

Phase 1: Generate a transaction database $\mathcal{DB}$. Apply DPextract with the extraction rules $\mathcal{R}$ and the seed opinion words $\mathcal{O}$ to each domain reviews $\mathcal{D}_i \in \mathcal{D}$ to extract a set of aspects. The set of resulting aspects for a domain $\mathcal{D}_i$ forms a transaction in $\mathcal{DB}$. From $\mathcal{D}$, we thus generate n transactions.

Phase 2: Apply an association rule mining algorithm to generate a set of rules with *minimum support* and *minimum confidence*. To prevent too many meaningless rules from being generated, those aspects appear in almost every review domain (we use 80% as the threshold) are removed from each transaction in $\mathcal{DB}$. For example, every review domain may have the aspect "price," i.e., it appears in almost every transaction in $\mathcal{DB}$ and can result in a huge number of useless association rules. Since our main goal is to use the generated association rules to recommend specific aspects rather than general aspects in every domain such as "price," the reduced $\mathcal{DB}$ rather than the original $\mathcal{DB}$ is used to produce a set $\mathcal{R}^a$ of association rules to be used for recommendation.

Recommending Aspects by AR-recom

Algorithm 3 gives the details of AR-recom($\mathcal{T}^-$, $\mathcal{T}$), which recommends aspects based on aspect association rules. For each association rule r in $\mathcal{R}^a$, if the antecedent of r is a subset of $\mathcal{T}^-$ (line 2), then recommend the terms in cons(r) $\cap$ $\mathcal{T}$ into the set $\mathcal{T}^a$ (line 3). The function ante(r) returns the set of terms in r's antecedent, and the function cons(r) returns the set of terms in r's consequent.

Algorithm 3 AR-recom($\mathcal{T}^-$, $\mathcal{T}$)

Input: Aspect sets $\mathcal{T}^-$ and $\mathcal{T}$
Output: Recommended aspect set $\mathcal{T}^a$
1: **for** each association rule $r \in \mathcal{R}^a$ **do**
2: **if** (ante(r) $\subseteq \mathcal{T}^-$) **then**
3: $\mathcal{T}^a \leftarrow \mathcal{T}^a \cup$ (cons(r) $\cap \mathcal{T}$);
4: **end if**
5: **end for**

For example, one association rule in $\mathcal{R}^a$ could be:

picture, display $\rightarrow$ video, purchase

whose antecedent contains "picture" and "display," and consequent contains "video" and "purchase." If both words "picture" and "display" are in $\mathcal{T}^-$, and only "video" is in $\mathcal{T}$, then we can only add "video" into $\mathcal{T}^a$.

Experiments

Experiment Settings

Here we introduce the experiment datasets, baseline approaches and evaluation metrics.

Dataset for Learning Word Vectors. To learn word vectors, we used the review collection created by (Jindal and Liu 2008), which contains more than 5.8 million Amazon reviews of all kinds of products. The open source tool word2vec[1] is employed to train our word vectors with its default parameter settings.

Datasets of Multiple Domains. To exploit existing extraction results from multiple past domains to mine association rules for recommendation, we used the review collection from 100 types of products created by (Chen and Liu 2014), consisting of 50 types of electronic products and 50 types of non-electronic products. Each product has 200 reviews.

Test Datasets for Evaluation. For aspect extraction evaluation, we need aspect-annotated data. The above two big review datasets are not annotated. We thus use two publicly available aspect-annotated corpora[2]. One is from (Hu and Liu 2004), which has been widely used in aspect extraction evaluation by researchers (Hu and Liu 2004; Popescu and Etzioni 2005; Qiu et al. 2011). This corpus has reviews of five products: 2 digital cameras (D1, D2), 1 cell phone (D3), 1 MP3 player (D4), and 1 DVD player (D5). The other is from (Liu et al. 2015), which contains reviews of three products: computer (D6), wireless router (D7), and speaker (D8). The details about these test datasets are given in Table 1.

Seed Opinion Words. We use all adjective opinion words in the lexicon of (Hu and Liu 2004)[2].

[1]https://code.google.com/archive/p/word2vec/

[2]http://www.cs.uic.edu/~liub/FBS/sentiment-analysis.html

Table 1: Detailed information of the test datasets.

Data	Product	# of Sentences	# of Aspects
D1	Digital camera	597	237
D2	Digital camera	346	174
D3	Cell phone	546	302
D4	MP3 player	1716	674
D5	DVD player	740	296
D6	Computer	531	354
D7	Wireless router	879	307
D8	Speaker	689	440

Compared Approaches. We compare the AER method with five baseline methods: DP, DP$^-$, DP$^+$, SimR, and ARR. We compare with DP as we use it as the base method, but our approach is independent of the base method, and can be used to improve other base extraction methods as well.

DP denotes the original DP method in (Qiu et al. 2011). It uses 8 extraction patterns, which can be expanded into rules after instantiating the relation variables with 8 dependency relations (*amod*, *prep*, *nsubj*, *csubj*, *xsubj*, *dobj*, *iobj* and *conj*). For dependency parsing, we used Stanford Parser[3].

DP$^-$ uses a set of high precision rules ($\mathcal{R}^-$ in Algorithm 1) selected from the DP rules based on their precisions. Each rule in DP$^-$ has a precision of at least 80% on D1 to D8.

DP$^+$ uses 8 extraction patterns in DP, but DP$^+$ uses more dependency relations ($\mathcal{R}^+$ in Algorithm 1). DP$^+$ uses 18 dependency relations, i.e., *amod*, *prep*, *nsubj*, *csubj*, *xsubj*, *dobj*, *iobj*, *conj*, *advmod*, *dep*, *cop*, *mark*, *nsubjpass*, *pobj*, *acomp*, *xcomp*, *csubjpass*, and *poss*. This results in a larger rule set. Clearly, DP$^+$ has a much higher extraction recall than DP and DP$^-$, but much lower precision.

SimR implements Algorithm 1 without line 5, it uses only aspect similarities for recommendation. There are various methods for similarity computing, we evaluate five of them, i.e., word vector, named as WV, which is described in the previous section, and four WordNet-based methods, i.e., SYN, which uses only WordNet synonyms to do recommendation, WUP (Wu and Palmer 1994), which calculates similarities by considering the depths of two synsets in WordNet, along with the depth of their least common subsume, LIN (Lin 1998), which uses the notion of information content in the form of conditional probability of encountering an instance of a child-synset given an instance of a parent synset, and PATH (Zhang, Gentile, and Ciravegna 2013), which computes the similarities by counting the number of nodes along the shortest path between the word senses in the 'is-a' hierarchies of WordNet.

ARR implements Algorithm 1 without line 4, i.e., it uses only aspect associations for recommendation. In this model, different sets of aspects can be used to generate association rules, resulting in different performances of ARR. We explore four groups of settings, each group consists of two settings, i.e., using the aspects extracted from 50 electronic domains to generate association rules, and using the aspects extracted from all 100 domains to generate association rules. The four group settings use the models DP$^-$, DP, DP$^+$ and SimR to extract aspects respectively.

AER implements Algorithm 1. Since SimR and ARR

[3]http://nlp.stanford.edu:8080/parser/

Table 2: Precision, Recall and F_1-score of the compared approaches evaluated based on multiple aspect term occurrences.

Data	DP			DP$^-$			DP$^+$			SimR			ARR			AER		
	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1
D1	70.7	91.0	79.6	84.8	66.8	74.8	66.2	96.3	78.5	82.4	87.7	85.0	89.0	73.7	80.6	80.5	89.8	84.9
D2	73.6	89.5	80.8	95.2	59.8	73.4	65.7	95.9	78.0	82.1	89.7	85.7	88.4	71.3	78.9	82.6	94.8	88.3
D3	76.5	90.2	82.8	85.7	54.8	66.9	65.6	95.1	77.6	86.4	86.2	86.3	91.7	67.3	77.6	86.5	87.2	86.9
D4	69.7	88.7	78.1	81.3	67.2	73.6	62.2	95.6	75.4	76.6	90.8	83.1	92.0	70.6	79.9	81.8	92.8	86.9
D5	63.0	89.6	74.0	88.9	63.7	74.2	58.8	94.3	72.4	87.0	82.9	84.9	91.5	78.5	84.5	88.0	88.5	88.2
Avg	70.7	89.8	79.1	87.2	62.5	72.6	63.7	**95.4**	76.4	82.9	87.5	85.0	**90.5**	72.3	80.4	83.9	90.6	**87.0**
D6	73.8	88.8	80.6	91.7	58.7	71.6	66.3	95.1	78.1	82.9	80.2	81.5	90.0	70.4	79.0	86.9	80.2	83.4
D7	65.5	91.6	76.4	67.6	45.4	54.3	55.8	97.4	70.9	74.2	83.3	78.5	86.2	73.6	79.4	73.0	92.3	81.5
D8	71.0	91.4	79.9	89.5	61.5	72.9	62.1	96.3	75.5	79.2	83.7	81.4	87.8	76.8	81.9	80.7	83.5	82.1
Avg	70.1	90.6	79.0	82.9	55.2	66.3	61.4	**96.2**	74.8	78.8	82.4	80.5	**88.0**	73.6	80.1	80.2	85.3	**82.3**

Table 3: Precision, Recall and F_1-score of the compared approaches evaluated based on distinct aspect terms.

Data	DP			DP$^-$			DP$^+$			SimR			ARR			AER		
	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1
D1	60.0	83.9	70.0	83.9	44.1	57.8	46.6	91.4	61.7	72.6	78.5	75.5	71.9	52.0	60.4	72.8	81.7	77.0
D2	59.6	78.8	67.9	93.8	34.3	50.3	46.3	89.4	61.0	70.9	80.6	75.4	70.0	51.1	59.1	72.7	86.6	79.1
D3	58.1	81.4	67.9	87.5	44.9	59.3	45.9	87.6	60.3	74.8	73.5	74.1	75.1	49.9	60.0	75.5	75.5	75.5
D4	53.9	74.7	62.6	77.2	42.4	54.7	46.1	88.0	60.5	65.4	78.2	71.2	68.8	48.4	56.8	68.8	80.7	74.3
D5	52.8	76.3	62.4	88.9	36.2	51.4	45.6	87.1	59.8	76.3	60.6	67.6	79.6	54.2	64.5	79.1	68.8	73.6
Avg	56.9	79.0	66.1	**86.2**	40.4	54.7	46.1	**88.7**	60.7	72.0	74.3	72.8	73.1	51.1	60.1	73.8	78.7	**75.9**
D6	63.4	78.5	70.1	90.5	43.1	58.4	52.2	88.5	65.6	74.6	67.7	71.0	80.1	55.3	65.4	83.1	69.4	75.6
D7	55.3	84.8	67.0	62.5	33.3	43.5	42.6	94.3	58.6	62.9	82.9	71.5	64.6	61.4	63.0	64.7	86.9	74.1
D8	56.5	80.8	66.5	86.7	43.7	58.1	44.2	90.7	59.5	66.1	72.9	69.3	76.3	55.8	64.5	69.7	70.3	70.0
Avg	58.4	81.3	67.9	**79.9**	40.0	53.3	46.3	**91.2**	61.2	67.9	74.5	70.6	73.7	57.5	64.3	72.5	75.5	**73.3**

have different settings, there are also different combinations of this model, which will be discussed later.

All approaches use Stanford Parser for dependency parsing. Note that, we extract not only single noun aspects but also multi-word expressions in all approaches. Noun phrases are identified based on the dependency relation *nn* (noun compound modifier) output by Stanford Parser. The identified noun phrases are treated as ordinary nouns in the rules.

Evaluation Metrics. Precision, recall, and F_1-score are used as our evaluation metrics. There are two ways to compute the results: (1) based on multiple occurrences of each aspect term, (2) based on distinct occurrence of each aspect term.

In a dataset, an important aspect often occurs many times, e.g., the aspect "picture" occurred 10 times in a set of camera reviews. For (1), if any occurrence of "picture" is extracted, then all occurrences of "picture" are considered extracted, i.e., 10. If none of its occurrences is extracted, it is considered as 10 losses. In (2), if any occurrence of "picture" is extracted, it is considered as one extraction. If none is extracted, it is considered as one loss. (2) clearly makes sense, but (1) also makes good sense because it is crucial to get those important aspects extracted. Extracting (or missing) a more frequent aspect term is rewarded (or penalized) more heavily than extracting (or missing) a less frequent one.

Let an extraction method return a set $\mathcal{A}$ of distinct aspect terms, and the set of distinct aspect terms labeled by human annotators be $\mathcal{T}$. TP (true positives) is $|\mathcal{A} \cap \mathcal{T}|$, FP (false positives) is $|\mathcal{A} \setminus \mathcal{T}|$, FN (false negatives) is $|\mathcal{T} \setminus \mathcal{A}|$.

For (2), the evaluation metrics are defined as:

$$P = \frac{TP}{TP+FP}, \quad R = \frac{TP}{TP+FN}, \quad F_1 = \frac{2 \times P \times R}{P+R}$$

For (1), F_1-score is computed in the same way, but precision and recall computations need change because we now consider multiple occurrences of the same aspect:

$$P = \frac{\sum_{i=1}^{|\mathcal{A}|} f_i \times E(a_i, \mathcal{A})}{\sum_{i=1}^{|\mathcal{A}|} f_i}, \quad R = \frac{\sum_{i=1}^{|\mathcal{T}|} f_i \times E(a_i, \mathcal{T})}{\sum_{i=1}^{|\mathcal{T}|} f_i}$$

where f_i is the term frequency of a_i, $E(a_i, \mathcal{A})$ (or $E(a_i, \mathcal{T})$) equals to 1 if a_i is an element of $\mathcal{A}$ (or $\mathcal{T}$), otherwise $E(a_i, \mathcal{A})$ (or $E(a_i, \mathcal{T})$) equals to 0.

Experimental Results

Table 2 shows the results of DP, DP$^-$, DP$^+$, SimR, ARR, and AER tested on datasets D1 to D8, based on multiple occurrences of aspect terms (evaluation (1))[4]. Table 3 shows the corresponding results of the approaches evaluated based on distinct aspect terms (evaluation (2)).

We empirically set the parameters for aspect similarities as ϵ = 0.38 in Algorithm 2, set the parameters for generating association rules as *minimum confidence* = 0.5 and *minimum support* = 0.2. Since the best SimR setting is WV, the best ARR setting is SimR(100), i.e., using the aspects extracted by SimR from 100 domains to generate association rules and then using the generated association rules to do recommendation, and the best AER combination is WV and SimR(100), the results in the tables for SimR, ARR and AER are from WV, SimR(100) and their combination. We will study these experiment settings later.

DP, DP$^-$, DP$^+$ and SimR are directly applied to D1 to D8. For ARR and AER, the extraction results from multiple domain datasets are employed first to generate association rules, then they are used to do recommendation on D1 to D8. Note that, DP, DP$^-$ and DP$^+$ cannot employ additional review datasets from other domains to help the extraction of

[4]The results of DP in Table 2 are different from those reported in (Qiu et al. 2011) due to two reasons. First, Minipar was used in (Qiu et al. 2011) but it is no longer available, so we use the Stanford Parser. (Qiu et al. 2011) revised some annotations in the datasets in their evaluation while we use the original annotations in evaluation.

the 8 evaluation datasets. That is exactly the problem that AER aims to address. As we will see from the results, AER can take good advantage of the additional review datasets.

From the tables, we observe that AER's F_1-scores are markedly better than those of DP, DP^- and DP^+, which demonstrate the effectiveness of our approach in recommending high quality aspects. For example, the important aspects "feature," "software" and "price" cannot be extracted by DP from D2, but they are recommended by AER. SimR's F_1-scores are also much better than those of DP, DP^- and DP^+, which means that aspect similarity, specifically, word vector-based similarity is very effective. ARR's F_1-scores are better than those of DP^-, and it has an average 13% recall improvement compared with DP^-, which means that aspect associations are important for aspect recommendation. The average precisions of ARR are better than those of DP^- in Table 2 because ARR can recommend some important aspects with high frequencies that are missed in DP^-. However, ARR's recalls are much lower than those of DP and DP^+, which means that using aspect associations is not enough for aspect recommendation.

We can also see that AER is better than SimR and ARR. This means that there are aspects that can be recommended based on aspect similarities but cannot be recommended based on aspect associations, and vice versa. For example, we can recommend "photo" as it is similar to the aspect "picture," which is extracted by DP^-, but it is not recommended based on aspect associations. We can recommend "color" because we have the following association rule:

$$\text{picture, image} \rightarrow \text{color}$$

where "picture" and "image" are extracted by DP^-, but "color" cannot be recommended by aspect similarities.

The average precision of DP is higher than that of DP^+ but lower than that of DP^-, and the average recall of DP is lower than that of DP^+ but higher than that of DP^-. This is because DP^+ uses more rules than DP due to the 10 additional dependency relations while DP^- uses less rules than DP as DP^- is only a subset of DP. The new rules in DP^+ bring more correct aspects (higher recall) but also more errors (lower precision), and the selected high precision rules in DP^- bring less aspects (lower recall) but more accurate ones (higher precision). Since the average F_1-score of DP is higher than those of DP^+ and DP^-, we can say that adding rules arbitrarily can harm the results for the existing approach in (Qiu et al. 2011).

In summary, our proposed approach AER can produce much better results than existing state-of-the-art rule-based approaches by employing additional knowledge extracted from large collections of unsupervised review datasets.

Experimental Setting Study

We show the performances of different experimental settings of SimR, ARR and AER in Figure 2 and Figure 3. All models are evaluated based on (1) multiple aspect occurrences, which are shown in sub-figures (a), and (2) distinct aspect occurrence, which are shown in sub-figures (b). The models are selected based on average F_1-scores on D1 to D8.

To show the effectiveness of different similarity measures in SimR, we compare the F_1-score of different SimR settings

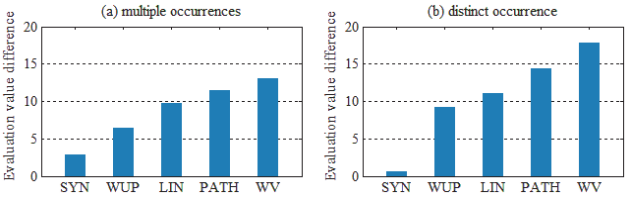

Figure 2: Average F_1-score differences between five SimR settings and DP^- on D1 to D8.

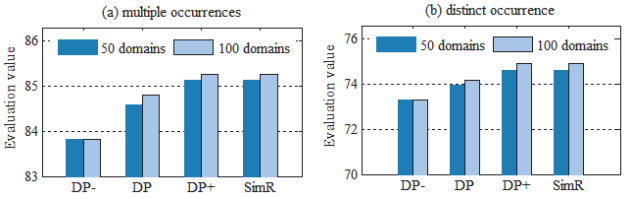

Figure 3: Average F_1-scores of AER on D1 to D8 as the method used to extract aspects for generating association rules changes. WV is used as the similarity measure in AER.

with DP^-. Figure 2 shows that the best SimR similarity measure is WV (word vector). It gets the biggest improvement. The improvement of SYN is small because the number of synonyms of each word is very small, and it also cannot recommend multi-word phrases such as "battery life" and "picture quality." Since the best similarity for SimR is WV, we use it in the final SimR and AER models.

Finally, we show how the results vary with different ARR settings, i.e., different approaches for producing the aspects from past domains used in association rule mining. Since the behavior of ARR is similar to that of AER, we only show the results of AER. Figure 3 gives the average F_1-scores of AER on D1 to D8 using the sets of aspects extracted by different approaches from multiple past domains. We first observe that using more product domains to generate association rules helps to improve AER's results. The first 50 domains are electronic products which have good aspect sharing with the 8 test datasets. It is interesting to see that adding the 50 non-electronic domains also helps to some extent. Using DP^+ and SimR to generate aspects in past domains for association rule mining produce similar best results.

Conclusion

This paper proposed a recommendation-based approach to improve the syntactic rule-based method for aspect extraction. Two forms of recommendation were presented. They exploit two types of interesting knowledge about aspects: aspect similarity and association. Such knowledge can be discovered from a large review corpus automatically. Experimental results demonstrated the superior performance of AER. In our future work, we plan to explore how the recommendation can also be used to improve other extraction methods. This is possible because it is flexible to choose different base extraction algorithms in AER.

Acknowledgments. Bing Liu's research was partially supported by the NSF grants IIS-1111092 and IIS-1407927, and

a gift award from Bosch. Yuanlin Zhang's work was partially supported by the NSF grants IIS-1018031 and CNS-1359359. Zhiqiang Gao's research was partially supported by the 863 program grant 2015AA015406 and the NSF of China grant 61170165.

References

Adomavicius, G., and Tuzhilin, A. 2005. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering* 17(6):734–749.

Agrawal, R., and Srikant, R. 1994. Fast algorithms for mining association rules in large databases. In *VLDB '94*, 487–499.

Brody, S., and Elhadad, N. 2010. An unsupervised aspect-sentiment model for online reviews. In *NAACL '10*, 804–812.

Chen, Z., and Liu, B. 2014. Mining topics in documents: Standing on the shoulders of big data. In *KDD '14*, 1116–1125.

Chen, Z.; Mukherjee, A.; and Liu, B. 2014. Aspect extraction with automated prior knowledge learning. In *ACL '14*, 347–358.

Choi, Y., and Cardie, C. 2010. Hierarchical sequential learning for extracting opinions and their attributes. In *ACL '10*, 269–274.

Collobert, R., and Weston, J. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *ICML '08*, 160–167.

Fang, L., and Huang, M. 2012. Fine granular aspect analysis using latent structural models. In *ACL '12*, 333–337.

Hu, M., and Liu, B. 2004. Mining and summarizing customer reviews. In *KDD '04*, 168–177.

Huang, E. H.; Socher, R.; Manning, C. D.; and Ng, A. Y. 2012. Improving word representations via global context and multiple word prototypes. In *ACL '12*, 873–882.

Jakob, N., and Gurevych, I. 2010. Extracting opinion targets in a single- and cross-domain setting with conditional random fields. In *EMNLP '10*, 1035–1045.

Jindal, N., and Liu, B. 2008. Opinion spam and analysis. In *WSDM '08*, 219–230.

Jo, Y., and Oh, A. H. 2011. Aspect and sentiment unification model for online review analysis. In *WSDM '11*, 815–824.

Joseph, T.; Lev, R.; and Yoshua, B. 2010. Word representations: A simple and general method for semi-supervised learning. In *ACL' 10*, 384–394.

Lafferty, J.; McCallum, A.; and Pereira, F. C. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML '01*, 282–289.

Li, F.; Huang, M.; and Zhu, X. 2010. Sentiment analysis with global topics and local dependency. In *AAAI '10*, 1371–1376.

Lin, C., and He, Y. 2009. Joint sentiment/topic model for sentiment analysis. In *CIKM '09*, 375–384.

Lin, D. 1998. An information-theoretic definition of similarity. In *ICML '98*, 296–304.

Liu, K.; Xu, L.; Liu, Y.; and Zhao, J. 2013. Opinion target extraction using partially-supervised word alignment model. In *IJCAI '13*, 2134–2140.

Liu, Q.; Gao, Z.; Liu, B.; and Zhang, Y. 2015. Automated rule selection for aspect extraction in opinion mining. In *IJCAI '15*, 1291–1297.

Liu, B. 2012. *Sentiment Analysis and Opinion Mining*. Morgan & Claypool Publishers.

Mei, Q.; Ling, X.; Wondra, M.; Su, H.; and Zhai, C. 2007. Topic sentiment mixture: Modeling facets and opinions in weblogs. In *WWW '07*, 171–180.

Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G. S.; and Dean, J. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems 26*. Curran Associates, Inc. 3111–3119.

Mitchell, M.; Aguilar, J.; Wilson, T.; and Van Durme, B. 2013. Open domain targeted sentiment. In *ACL '13*, 1643–1654.

Mnih, A., and Hinton, G. E. 2009. A scalable hierarchical distributed language model. In *NIPS '09*. 1081–1088.

Moghaddam, S., and Ester, M. 2011. ILDA: interdependent lda model for learning latent aspects and their ratings from online product reviews. In *SIGIR '11*, 665–674.

Mukherjee, A., and Liu, B. 2012. Aspect extraction through semi-supervised modeling. In *ACL '12*, volume 1, 339–348.

Pavlopoulos, J., and Androutsopoulos, I. 2014. Multi-granular aspect aggregation in aspect-based sentiment analysis. In *EACL '14*, 78–87.

Pennington, J.; Socher, R.; and Manning, C. D. 2014. Glove: Global vectors for word representation. In *EMNLP '14*, 1532–1543.

Popescu, A.-M., and Etzioni, O. 2005. Extracting product features and opinions from reviews. In *HLT-EMNLP '05*, 339–346.

Qiu, G.; Liu, B.; Bu, J.; and Chen, C. 2011. Opinion word expansion and target extraction through double propagation. *Computational Linguistics* 37(1):9–27.

Titov, I., and McDonald, R. 2008. A joint model of text and aspect ratings for sentiment summarization. In *ACL '08: HLT*, 308–316.

Wang, B., and Wang, H. 2008. Bootstrapping both product features and opinion words from chinese customer reviews with cross-inducing. In *IJCNLP '08*, 289–295.

Wang, H.; Lu, Y.; and Zhai, C. 2010. Latent aspect rating analysis on review text data: A rating regression approach. In *KDD '10*, 783–792.

Wu, Z., and Palmer, M. 1994. Verbs semantics and lexical selection. In *ACL '94*, 133–138.

Wu, Y.; Zhang, Q.; Huang, X.; and Wu, L. 2009. Phrase dependency parsing for opinion mining. In *EMNLP '09*, 1533–1541.

Xu, L.; Liu, K.; and Zhao, J. 2014. Joint opinion relation detection using one-class deep neural network. In *COLING '14*, 677–687.

Zhang, L.; Liu, B.; Lim, S. H.; and O'Brien-Strain, E. 2010. Extracting and ranking product features in opinion documents. In *COLING '10: Posters*, 1462–1470.

Zhang, Z.; Gentile, A. L.; and Ciravegna, F. 2013. Recent advances in methods of lexical semantic relatedness - a survey. *Natural Language Engineering* 19(4):411–479.

Zhao, W. X.; Jiang, J.; Yan, H.; and Li, X. 2010. Jointly modeling aspects and opinions with a maxent-lda hybrid. In *EMNLP '10*, 56–65.

Zhou, X.; Wan, X.; and Xiao, J. 2013. Collective opinion target extraction in Chinese microblogs. In *EMNLP '13*, 1840–1850.

Zhu, J.; Wang, H.; Tsou, B. K.; and Zhu, M. 2009. Multi-aspect opinion polling from textual reviews. In *CIKM '09*, 1799–1802.

Zhuang, L.; Jing, F.; and Zhu, X.-Y. 2006. Movie review mining and summarization. In *CIKM '06*, 43–50.