ExtRA: Extracting Prominent Review Aspects from Customer Feedback

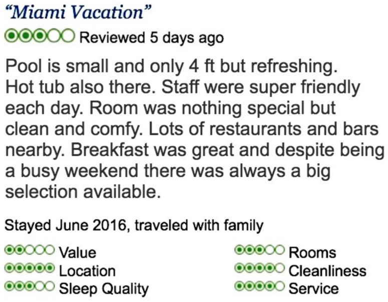
Zhiyi Luo, Shanshan Huang, Frank F. Xu
Bill Yuchen Lin, Hanyuan Shi, Kenny Q. Zhu
Shanghai Jiao Tong University, Shanghai, China
{jessherlock, huangss_33}@sjtu.edu.cn, frankxu2004@gmail.com
{yuchenlin, shihanyuan}@sjtu.edu.cn, kzhu@cs.sjtu.edu.cn

Abstract

Many existing systems for analyzing and summarizing customer reviews about products or service are based on a number of prominent review aspects. Conventionally, the *prominent review aspects* of a product type are determined manually. This costly approach cannot scale to large and cross-domain services such as Amazon.com, Taobao.com or Yelp.com where there are a large number of product types and new products emerge almost everyday. In this paper, we propose a novel framework, for extracting the most prominent aspects of a given product type from textual reviews. The proposed framework, ExtRA, extracts K most prominent aspect terms or phrases which do not overlap semantically automatically without supervision. Extensive experiments show that ExtRA is effective and achieves the state-of-the-art performance on a dataset consisting of different product types.

1 Introduction

Online user review is an essential part of e-commerce. Popular e-commerce websites feature an enormous amount of text reviews, especially for popular products and services. To improve the user experience and expedite the shopping process, many websites provide qualitative and quantitative analysis and summary of user reviews, which is typically organized by different *prominent review aspects*. For instance, Figure 1 shows a short review passage from a customer on TripAdvisor.com, and the customer is also asked to give scores on several specific aspects of the hotel, such as *location* and *cleanness*. With aspect-based reviews summary, potential customers can assess a product from various essential aspects very efficiently and directly. Also, aspect-based review summary offers an effective way to group products by their prominent aspects and hence enables quick comparison.



"Miami Vacation"
●●●○○ Reviewed 5 days ago

Pool is small and only 4 ft but refreshing. Hot tub also there. Staff were super friendly each day. Room was nothing special but clean and comfy. Lots of restaurants and bars nearby. Breakfast was great and despite being a busy weekend there was always a big selection available.

Stayed June 2016, traveled with family

●●○○○ Value ●●●○○ Rooms
●●●●● Location ●●●●○ Cleanliness
●●●○○ Sleep Quality ●●●●○ Service

Figure 1: An example user review about a hotel on TripAdvisor. The grades are organized by different prominent review aspects: *value*, *rooms*, etc.

Existing approaches for producing such prominent aspect terms have been largely manual work (Poria et al., 2014; Qiu et al., 2011). This is feasible for web services that only sell (or review) a small number of product types of the same domain. For example, TripAdvisor.com only features travel-related products, and Cars.com only reviews automobiles, so that human annotators can provide appropriate aspect terms for customers based on their domain knowledge. While it is true that the human knowledge is useful in characterizing a product type, such manual approach does not scale well for general-purpose e-commerce platforms, such as Amazon, eBay, or Yelp, which feature too many product types, not to mention that new product and service types are emerging everyday. In these cases, manually selecting and pre-defining aspect terms for each type is too costly and even impractical.

Moreover, the key aspects of a product type may also change over time. For example, in the past, people care more about the screen size and signal intensity when reviewing cell phones. These aspects are not so much of an issue in present days. People instead focus on battery life and processing speed, etc. Therefore, there is a growing need to automatically extract prominent aspects from user reviews.

A related but different task is *aspect-based opinion mining* (Su et al., 2008; Zeng and Li, 2013). Here techniques have been developed to automatically mine product-specific "opinion phrases" such as those shown in Figure 2. In this example, the most frequently mentioned opinion phrases about a phone model along with the mention frequency are displayed. Their goal is to get the *fine-grained* opinion summary on possibly overlapping aspects of a particular product. For example, "good looks" and "beautiful screen" both comments on the "appearance" aspect of the phone. However, these aspects are implicit and can't be used in aspect-based review summarization directly. The main disadvantage of these opinion phrases is that their aspects differ from product to product, making it difficult to compare the product side by side.

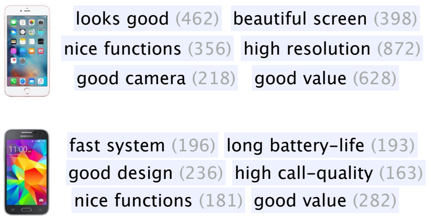


Figure 2: Automatic review summarization for two mobile phones on an e-commerce website

The goal of this paper is to develop an unsupervised framework for automatically extracting K most prominent, non-overlapping review aspects for a given type of product from user review texts. Developing such an unsupervised framework is challenging for the following reasons:

- The extracted prominent aspects not only need to cover as many customer concerns as possible but also have little semantic overlap.
- The expression of user opinions is highly versatile: aspect terms can be expressed either explicitly or implicitly. For example, the mention of "pocket" implies the aspect "size".
- Product reviews are information rich. A short piece of comments may target multiple aspects, so topics transit quickly from sentence to sentence.

Most previous unsupervised approaches for the prominent aspect extraction task are variants of topic modeling techniques (Lakkaraju et al., 2011; Lin and He, 2009; Wang et al., 2011a). The main problem of such approaches is that they typically use only word frequency and co-occurrence information, and thus degrade when extracting aspects from sentences that appear different on the surface but actually discuss similar aspects.

Given all review text about a certain product type, our framework, ExtRA, extracts most prominent aspect terms in four main steps: first it extracts potential aspect terms from text corpus by lexico-syntactic analysis; then it associates the terms to synsets in WordNet and induce a sub-graph that connect these terms together; after that it ranks the aspect terms by a personalized page rank algorithm on the sub-graph; and finally picks the top K non-overlapping terms using the subsumption relation in the subgraph.

The main contributions in this paper are as follows:

1. We propose a novel framework for extracting prominent aspects from customer review corpora (Section 2), and provide an evaluation dataset for future work in this research area.
2. Extensive experiments show that our unsupervised framework is effective and outperforms the state-of-the-art methods by a substantial margin (Section 3).

2 Framework

In this section, we first state the *review aspect extraction problem*, then present the workflow of our method, shown in Figure 3.

2.1 Problem Statement

The review aspect extraction problem is given all the text reviews about one type of product or service, extract K words (or phrases), each of which represents a prominent and distinct review aspect.

For instance, if the given product type is *hotel*, we expect a successful extraction framework to extract $K = 5$ aspect terms as follows: *room, location, staff, breakfast, pool*.

2.2 Aspect Candidates Extraction

Following the observation of Liu (2004; 2015), we assume that aspect terms are nouns and noun phrases. First, we design a set of effective syntactic rules, which can be applied across domains, to collect the aspect candidates from review texts.

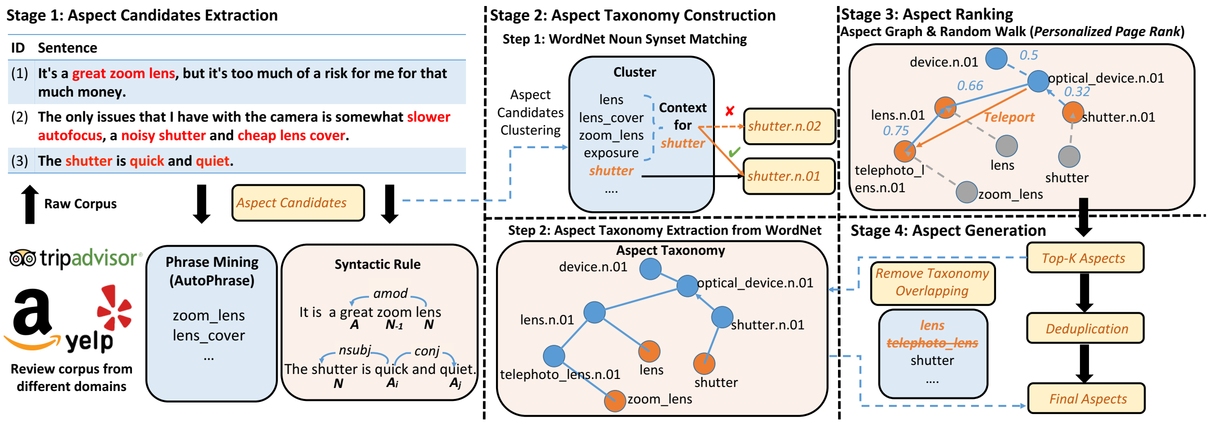

Stage 1: Aspect Candidates Extraction
ID Sentence
(1) It's a great zoom lens, but it's too much of a risk for me for that much money.
(2) The only issues that I have with the camera is somewhat slower autofocus, a noisy shutter and cheap lens cover.
(3) The shutter is quick and quiet.
Raw Corpus
Aspect Candidates
tripadvisor
yelp
Review corpus from different domains
Phrase Mining (AutoPhrase)
zoom_lens
lens_cover
...
Syntactic Rule
amod
It is a great zoom lens
A N-1 N
nsubj
conj
The shutter is quick and quiet.
N Ai Aj
Stage 2: Aspect Taxonomy Construction
Step 1: WordNet Noun Synset Matching
Aspect Candidates Clustering
Cluster
lens
lens_cover
zoom_lens
exposure
shutter
....
Context for shutter
shutter.n.02
shutter.n.01
Step 2: Aspect Taxonomy Extraction from WordNet
Aspect Taxonomy
device.n.01
optical_device.n.01
lens.n.01
shutter.n.01
telephoto_lens.n.01
lens
shutter
zoom_lens
Stage 3: Aspect Ranking
Aspect Graph & Random Walk (Personalized Page Rank)
device.n.01
0.5
optical_device.n.01
0.66
0.32
lens.n.01
0.75
Teleport
shutter.n.01
lens
shutter
telephoto_l ens.n.01
zoom_lens
Stage 4: Aspect Generation
Top-K Aspects
Remove Taxonomy Overlapping
lens
~~telephoto_lens~~
shutter
....
Deduplication
Final Aspects

Figure 3: Overall framework.

We mainly use the adjectival modifier dependency relation (*amod*) and the nominal subject relation (*nsubj*) to extract the aspect-opinion pairs $\langle N, A \rangle$. In addition, we leverage the conjunction relation (*conj*) between adjectives to complement the extracted pairs. Formally, the extraction rules can be specified as follows:

Rule 1. If $amod(N, A)$, then extract $\langle N, A \rangle$.

Rule 2. If $nsubj(A, N)$, then extract $\langle N, A \rangle$.

Rule 3. If $\langle N, A_i \rangle$ and $conj(A_i, A_j)$, then extract $\langle N, A_j \rangle$.

In this case, N indicates a noun, and A (e.g. A_i, A_j) is an adjective. The dependencies (e.g. $amod(N, A)$) are expressed as *rel(head, dependent)*, where *rel* is the dependency relation which holds between *head* and *dependent*. Note that many aspects are expressed by phrases, thus, we extend the phrases as aspect candidates by introducing the extension rules as follows:

Rule E1. If $\langle N, A \rangle$ and $N_{-1}\_N \in P$, then use $\langle N_{-1}\_N, A \rangle$ to replace $\langle N, A \rangle$.

Rule E2. If $\langle N, A \rangle$ and $N\_N_{+1} \in P$, then use $\langle N\_N_{+1}, A \rangle$ to replace $\langle N, A \rangle$.

where N_{-1} and N_{+1} denotes the noun word, and the subscript represents displacement to N in the sentence. We use AutoPhrase (Liu et al., 2017) to extract a set of phrases P with high coherence. Then we use P to filter out the incoherent phrases so as to obtain the high-quality phrases as aspect candidates. The example in Figure 3 (Stage 1) demonstrates the extraction process. For example, we extract the pair ⟨ *great*, *zoom_lens* ⟩ from sentence (1) by applying *Rule 1* and *Rule E1*. Similarly, the extraction rules match ⟨ *slower*, *autofocus* ⟩, ⟨ *noisy*, *shutter* ⟩, ⟨ *cheap*, *lens_cover* ⟩ in sentence (2) and ⟨ *quick*, *shutter* ⟩, ⟨ *quiet*, *shutter* ⟩ in sentence (3) as potential aspect-opinion pairs. After extracting such pairs from the text reviews, we sort them by the number of occurrences, and extract the nouns and noun phrases in the top pairs as aspect candidates, assuming that the most prominent aspects are subsumed by those candidates terms.

2.3 Aspect Taxonomy Construction

The aspect candidates extracted in the last stage come with the counts modified by adjectives. We can directly use such raw counts to rank the aspect candidates. This is one of the baseline models in our experiments. However, such ranking usually suffers from the aspect overlapping problem which obviously violates the principle of pursuing both coverage and distinctiveness of prominent aspects. For example, given the number of prominent aspects K as 5, we can extract both of 'location' and 'place' aspects from the hotel reviews. In order to solve this problem, we construct an aspect taxonomy to obtain such overlapping information between aspect candidates by leveraging the WordNet ontology.

2.3.1 WordNet Synset Matching

First, we need to match our aspect candidates onto WordNet synsets. The accuracy of synset matching is very important for our aspect taxonomy construction. This is actually a classical word sense disambiguation (WSD) problem. Our initial attempt is to use a Python WSD tool (Tan, 2014). For each aspect candidate, we take it as the target and randomly sample a bunch of sentences that contain this target. We use the extended word sense disambiguation algorithm (Banerjee and Pedersen, 2003) in this tool. We count the total occurrences for each noun sense (synset) of the candidate and match the candidate to the most frequent synset. However, such a method is not good enough for our problem, as shown in the results later. It only considers the local context information within the review sentence. Whats more, the review sentences are usually very short and colloquial, which makes it more difficult to match properly by a common WSD algorithm. Therefore, it is critical to construct more reliable contexts for aspect candidate matching.

To achieve this goal, we cluster the aspect candidates with similar semantics together. Then, for each aspect candidate, we take the other candidates within the same cluster as its context for later disambiguation. As shown in the first step of stage 2 in Figure 3, the semantic similar aspect candidates such as *lens, lens_cover, zoom_lens, exposure and shutter* are clustered together. For example, we can disambiguate the sense of *shutter* by leveraging *lens, lens_cover, zoom_lens, and exposure*. We observed that our aspect candidates can be fine-grain clustered with a two-stage k-means clustering method,[1] which generates the better context for the aspect candidates. More specifically, for a particular aspect candidate a_t from the cluster $C = \{a_1, a_2, ..., a_t, ..., a_n\}$, we calculate the context vector of a_t as:

$$c(a_t) = \sum_{i=1, i \neq t}^{n} E(a_i), \tag{1}$$

where $c(a_t)$ denotes the context vector of a_t, and $E(a_i)$ represents the embedding of a_i. The set of candidate synsets $S(a_t) = \{s_1^t, s_2^t, ..., s_m^t\}$ consists of the noun senses (e.g. s_i^t) of a_t from WordNet. Each sense s_i^t is associated with a *gloss* g_i^t (i.e. a brief definition of s_i^t) which covers the semantics of the sense. Therefore, we encode s_i^t as the summation of the word vectors in g_i^t:

$$v(s_i^t) = \sum_{j=1}^{q} E(w_j^{t,i}), \tag{2}$$

$W(g_i^t)$ is the sequence of words in g_i^t, i.e., $W(g_i^t) = [w_1^{t,i}, w_2^{t,i}, ..., w_q^{t,i}]$. For each candidate sense s_i^t of the aspect candidate a_t, we calculate the cosine semantic similarity between $v(s_i^t)$ and $c(a_t)$, and match a_t to the most similar s_i^t.

2.3.2 Aspect Taxonomy Extraction from WordNet

In order to construct the aspect taxonomy from WordNet, we first extract the hypernym paths for every matched synsets in the previous step. By definition, a hypernym path p of synset s is the is-a relation path from s to the root synset (i.e. entity.n.01 for nouns). We extract the hypernym paths for each matched synset s_i in the WordNet ontology. Next, we scan over all the collected paths once to construct the aspect taxonomy which is a directed acyclic graph (DAG). In p, s_1 is the synset matched from our potential aspects, and s_{i+1} is the hypernym of s_i. As shown in step 2 of Stage 2 in Figure 3, we match the aspect candidate *shutter* to *shutter.n.01*. The only one hypernym path of *shutter.n.01* is *[shutter.n.01, optical_device.n.01, device.n.01, ..., entity.n.01]*.

However, the matched synset usually has multiple hypernym paths in WordNet. We use the following strategy to compact and minimize the aspect taxonomy:

- Among all the paths from an aspect candidate s_1, we will keep those paths that contain more than 1 aspect candidates, unless there's only one path from s_1. If all paths contain only 1 aspect candidate s_1 each, we will keep all of them.

- To further optimize the taxonomy structure, we induce a minimum subgraph from the original taxonomy using a heuristic algorithm (Kou et al., 1981). Such a subgraph satisfies the following conditions: 1) it contains all the nodes matched from aspect candidates; 2) the total number of nodes in the graph is minimal. Consequently, the induced graph is a weakly connected DAG.

After acquiring the aspect taxonomy for the given product or service, we can now tell if two aspects are semantically overlapped or not.

2.4 Aspect Ranking

In this section, we propose a novel method based on personalized page rank to compute the overall rank values for the potential aspects by leveraging the aspect taxonomy.

[1] The implementation details are in Section 3.2.

Let the aspect taxonomy be a graph $G = (V, E)$. Each node $v \in V$ is a synset in the aspect taxonomy and encoded as a vector by instantiating E as Glove embeddings in (2) . Each edge $e = \langle u, v \rangle$ carries a weight which is the semantic similarity between the nodes u and v, computed using cosine similarity.

Next, we perform the random walks on our constructed aspect taxonomy. The propagation starts from candidate aspect nodes in the aspect taxonomy, which are called *seeds* here. The rank values (aspect importance) of all nodes are:

$$x_t = (1 - \alpha) * Ax_{t-1} + \alpha * E, \tag{3}$$

where t is the time step in random walk process. In the initial state E(i.e. x_0), the aspect importance only distributes on the seeds ($v \in V_b$). E_i is the i-th dimension of E, indicating the portion of aspect importance on node s_i at time step 0. E is calculated as follows:

$$E_i = \begin{cases} \frac{f(le(s_i))}{\sum_{j=1}^{n} f(le(s_j))} & \text{, if } s_i \text{ is a seed} \\ 0 & \text{, otherwise,} \end{cases} \tag{4}$$

where n is the number of nodes in the graph, s_i is the synset node, $le(s_i)$ denotes the lemma form of s_i, and $f(le(s_i))$ represents the frequency that $le(s_i)$ is modified by adjectives.

The aspect importance is updated using the transition probabilities matrix A which are the normalized weights on the edges of the taxonomy. α is the teleport probability, which is the probability of returning to the initial distribution at each time step. α determines the distance of propagation of the taxonomy.

2.5 Aspect Generation

Finally, we generate the prominent aspects using the rank values of the aspects as well as the is-a relations in the aspect taxonomy.

We sort $le(s_i)$ in decreasing order by their rank values. We essentially take the top aspects from the sorted list. However there might be two types of overlapping that we need to avoid: i) duplicate: different synset nodes may map to the same aspects, i.e., $le(s_i) = le(s_j), s_i \neq s_j$ (aspects); ii) taxonomy overlap: the later aspect in the list is the hypernym or hyponym of the one of previous aspects. To this end, we just skip overlapped aspect, and move along the list until we generate K non-overlapping prominent aspects from the list.

3 Experiments

We compare the ExtRA framework with multiple strong baselines on extracting aspect terms from user reviews. We first introduce the dataset and the competing models, then show the quantitative evaluation as well as qualitative analysis for different models.

3.1 Dataset

We use the customer review corpora of 6 kinds of product and service [2] collected from popular websites, including Amazon, TripAdvisor and Yelp. The number of hotel reviews (Wang et al., 2011b) in the original corpus is huge. Therefore, we randomly sample 20% of the reviews to perform our experiments. The statistics of the corpora are shown in Table 1.

Table 1: Dataset statistics.

Product type	Source	#Reviews
hotel	TripAdvisor	3,155,765
mobile phone	Amazon	185,980
mp3 player	Amazon	30,996
laptop	Amazon	40,744
cameras	Amazon	471,113
restaurant	Yelp	269,000

Existing published aspect extraction datasets (Hu and Liu, 2004; Popescu and Etzioni, 2007; Pavlopoulos and Androutsopoulos, 2014; Ding et al., 2008) include only fine-grained aspects from reviews, which are not suitable for evaluating the performance of prominent aspects extraction. Therefore, we build a new evaluation dataset particularly for this task. Following the previous work (Ganu et al., 2009; Brody and Elhadad, 2010; Zhao et al., 2010; Wang et al., 2015) as well as the popular commercial websites (e.g. TripAdvisor), which most manually labeled 3-6 prominent aspects for rating, we set K as five. Therefore, we ask each annotator who are familiar with the domain to give 5 aspect terms which they think are most important for each category. We have five annotators in total. [3] One prominent aspect can be expressed by different terms. Thus, it is difficult to achieve a satisfied inner-agreement. We propose two evaluation methods, especially the soft accuracy in Section 3.3.1 to compensate this problem. To acquire a relatively higher inner-agreement, we educate the annotators with top 100

[2] The data is available from http://times.cs.uiuc.edu/~wang296/Data/ and https://www.yelp.com/dataset

[3] The complete labeled set of ExtRA is released at http://adapt.seiee.sjtu.edu.cn/extra/.

frequent aspect candidates as hints. Though, they are not required to pick up labels from the candidates. The inter-annotator agreement of each product type shown in Table 3 is computed as the average jaccard similarity between every two annotators.

3.2 Baselines and ExtRA

We introduce three topic modeling based baselines for the task. These are **LDA** (Blei et al., 2003), **BTM** (Cheng et al., 2014) and **MG-LDA** (Titov and McDonald, 2008). MG-LDA is a strong baseline which attempts to capture multi-grain topics (i.e. global & local), where the local topics correspond to the rateable prominent aspects. We treat each review as a document and perform those models to extract K topics. Then, we select most probable words in each topic as our extracted aspect terms. To prevent extracting the same aspects (w) from different topics, we only keep w for the topic t with the highest probability $p(w|t)$ value, then re-select aspects for the other topics until we get K different aspects. For fair comparison among different models, the number of target aspects K is set as 5. The hyper-parameter of MG-LDA (global topics) is set to 30 with fine-tuning.

Another syntactic rule-based baseline model **AmodExt** is from the first stage of our framework. After extracting the aspect candidates using *amod-rule* in Section 2.2, we sort the aspect candidates by their counts of extracted occurrences. Then select the top K candidates as the prominent aspects.

ABAE (He et al., 2017) is a neural based model that can to infer K aspect types. Each *aspect type* is a ranked list of representative words. To generate K prominent aspects, we first infer K aspect types using *ABAE*, then select the most representative word from each aspect type.

For **ExtRA**, in the taxonomy construction stage, we use a two-stage K-means clustering method for synset matching task, and the cluster number is auto-tuned using silhouette score (Rousseeuw, 1987). We use SkipGram (Mikolov et al., 2013) model to train the embeddings on review texts for k-means clustering. We set the dimension of the embeddings as 100 and run 64 epochs for each product corpora. In the aspect ranking stage, we empirically set the teleport probability α as 0.5 which indicates that the expected walk-length from the seeds is $\frac{1}{\alpha} = 2$.

Table 2: WordNet Synset matching accuracies

	hotel	mp3	cameras	mobile phone	laptop	restaurant
LESK	0.71	0.59	0.62	0.64	0.53	0.65
Cluster	**0.86**	**0.83**	**0.74**	**0.80**	**0.78**	**0.69**

Table 3: Inner-annotator agreements

	hotel	mp3	cameras	mobile phone	laptop	restaurant
Jaccard	0.470	0.554	0.304	0.440	0.271	0.671

3.3 Evaluation

In this section, we compare ExtRA with five baseline models both quantitatively and qualitatively.

3.3.1 Quantitative Evaluation

First, we perform two experiments to justify our aspect taxonomy construction stage:

- To justify the synset matching step, we compare our proposed cluster method with classical WSD algorithm (Lesk) on matching accuracy. We manually label 100 sampled synset nodes for each category. The synset matching accuracies are shown in Table 2. We can see that our clustering method is effective for the synset matching task.

- We induce the aspect taxonomy using a heuristic algorithm to obtain more compact and aspect-oriented subgraph. We show the size of aspect taxonomy induced before and after taxonomy minimization in Figure 4.

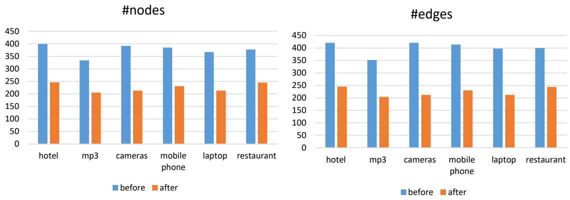


Figure 4: Statistics of induced aspect taxonomy before and after taxonomy minimization

Next, we evaluate our model as well as above baselines on the evaluation dataset described above. We did not remove the duplicate aspect labels for the qualitative evaluation, since the repeated aspects are assume to be better. For a given category, we first calculate the percentage of the 25 labels that exactly match one of the 5 aspect terms generated by the model as the *hard accuracy* of the model. Formally, $Aspects(m) =$

$[a_1, a_2, a_3, a_4, a_5]$ denotes the five prominent aspects generated from model m for the given category. $L = [l_1, l_2, ..., l_{25}]$ are the 25 golden aspect terms, where $L^{(h)} = [l_{5h-4}, ..., l_{5h}]$ are from the h-th human annotator. The hard accuracy is defined as:

$$hacc(m) = \frac{\sum_{i=1}^{25} hit(Aspects(m), l_i)}{25} \tag{5}$$

$$hit(Aspects(m), l_i) = \begin{cases} 1, & l_i \in Aspects(m) \\ 0, & \text{otherwise,} \end{cases} \tag{6}$$

However, counting the number of exact matches makes the accuracy score discrete and coarse. Besides, it penalizes aspect terms that don't match the label but actually have similar meanings. To remedy this, we propose the *soft accuracy* evaluation measure. For each set of five golden labels from h-th annotator, we first align each generated aspect $a_k \in Aspects(m)$ with one golden aspect $l_j \in L^{(h)}$ (i.e. $align^{(h)}(a_k) = l_j$). We align the exact match terms together, and then choose the optimal alignment for the others by permuting all possible alignments. The optimal alignment $align^{(h)}(a_k)$ acheives maximum soft accuracy. Then we calculate the soft matching score between $Aspects(m)$ and $L^{(h)}$ as $\sum_{k=1}^{K} sim(a_k, align^{(h)}(a_k))$, where sim is the cosine similarity computed by Glove (2014) [4]. We then compute the soft accuracy measure as follows:

$$sacc(m) = \frac{1}{5} * \sum_{h=1}^{5} \sum_{k=1}^{K} sim(a_k, align^{(h)}(a_k)), \tag{7}$$

where $K = 5$ in this case. The comparison results are shown in Table 4.

Our model (ExtRA) outperforms all the other baselines in all categories except cameras using the hard accuracy measure. Besides, ExtRA is the best model on four out of six products under the soft accuracy measure. As shown in Table 2, the accuracy for synset matching is relatively low for cameras and restaurant, resulting in the lower accuracy in overall aspect extraction.

Table 4: Comparison of *hard* (upper row) & *soft* (lower row) accuracies using different models for aspect extraction.

	LDA	BTM	MG-LDA	ABAE	AmodExt	ExtRA
hotel	0.16	0.16	0.16	0.16	0.44	**0.56**
	0.50	0.49	0.67	0.35	0.65	**0.70**
mp3	0.0	0.08	0.08	0.0	0.35	**0.44**
	0.47	0.49	0.47	0.32	0.58	**0.60**
camera	0.24	**0.40**	0.28	0.04	0.04	0.32
	0.56	**0.69**	0.54	0.29	0.41	0.55
mobile phone	0.16	0.0	0.28	0.0	0.52	**0.60**
	0.58	0.33	0.58	0.31	**0.73**	0.71
laptop	0.08	0.24	0.24	0.0	0.24	**0.28**
	0.40	0.50	0.50	0.22	0.51	**0.53**
restaurant	0.20	0.0	0.0	0.0	**0.56**	**0.56**
	0.49	0.38	0.42	0.29	**0.77**	0.72

Table 5: The five prominent aspect terms

hotel	LDA	room, pool, stay, good, nice
	BTM	walk, good, room, stay, check
	MGLDA	room, stay, good, location, staff
	ABAE	shouted, room, terrific, accommodation, alexanderplatz
	AmodExt	room, *location, place*, view, staff
	ExtRA	**room, location, view, staff, service**
mp3	LDA	work, great, good, music, ipod
	BTM	battery, ipod, work, song, good
	MGLDA	battery, ipod, music, song, good
	ABAE	documentation, content, portability, bought, table
	AmodExt	drive, quality, sound, feature, device
	ExtRA	**drive, sound_quality, feature, screen, software**
cameras	LDA	lens, picture, buy, video, mode
	BTM	battery, picture, function, lens, good
	MGLDA	battery, picture, good, mpcture, mode
	ABAE	toy, picture, mailed, ultrazoom, sharpness
	AmodExt	*picture, photo*, quality, feature, shot
	ExtRA	**image_quality, photograph, feature, shot, lens**
mobile phone	LDA	battery, buy, good, apps, work
	BTM	core, good, work, para, apps
	MGLDA	work, battery, screen, good, card
	ABAE	cracked, amazing, continuously, archive, bought
	AmodExt	feature, screen, price, camera, quality
	ExtRA	**feature, price, screen, quality, service**
laptop	LDA	screen, good, buy, drive, chromebook
	BTM	windows, screen, work, drive, good
	MGLDA	windows, battery, screen, good, year
	ABAE	salign, returned, affordable, downloads, position
	AmodExt	drive, machine, price, screen, life
	ExtRA	**drive, price, screen, deal, performance**
restaurant	LDA	food, good, room, time, great
	BTM	good, room, pour, time, order
	MGLDA	great, good, place, time, make
	ABAE	jones, polite, told, chickpea, place
	AmodExt	food, service, place, experience, price
	ExtRA	**service, food, experience, company, price**

3.3.2 Qualitative Analysis

To qualitatively evaluate different models, we present the extracted 5 aspect terms by each model from each domain in Table 5. Our model (ExtRA) has significant advantage over other baselines for that we can do better aspect extraction with reasonable results, and extract not only words but also phrases as prominent aspects, *e.g. sound quality, image quality*. The proposed model avoid the overlapping aspects appeared in our strong baseline (AmodExt) by deduplication using generated aspect taxonomy information. The overlapping aspects are marked in italics. For example, both *location* and *place* are extracted as top aspects, but they mean nearly the same concept. The results from other baseline methods, inevitably contain some sentiment words and opinions, like *good,*

[4] We use the GloVe embeddings with 300 dimensions, trained from 840B tokens using common crawl data.

nice, great, etc. Our model resolves such drawback by extracting aspect candidates from only nouns and using syntactic rules to find words that are frequently modified by adjectives.

4 Related Work

Existing research on *aspect-based review analysis* has focused on mining opinion based on given aspects (Su et al., 2008; Zeng and Li, 2013) or jointly extracting the aspects and sentiment (Lin and He, 2009; Zhao et al., 2010; Qiu et al., 2011; Wang et al., 2015; Liu et al., 2016). They are mostly interested in detecting aspect words in a given sentence, whereas our goal is to extract the most prominent aspects of a type of product from a large number of reviews about that product type. We divide the existing work on review aspect extraction into three types:

- *rule-based* methods, most of which utilize handcrafted rules to extract candidate aspects and then perform clustering algorithm on them.
- *topic modeling based* methods, which directly model topics from texts and then extract aspects from the topics.
- *neural network based* methods, which takes advantage of the recent deep neural network models.

4.1 Rule-based Methods

These methods leverage word statistical and syntactic features to manually design rules, recognizing aspect candidates from texts. Poria et al. (2014) use manually crafted mining rules. Qiu et al. (2011) also used rules, plus the Double Propagation method to better relate sentiment to aspects. Gindl et al. (2013) cooperate the Double Propagation with anaphora resolution for identifying co-references to improve the accuracy. Su et al. (2008) used a clustering method to map the implicit aspect candidates (which were assumed to be the noun form of adjectives in the paper) to explicit aspects. Zeng et al. (2013) mapped implicit features to explicit features using a set of sentiment words and by clustering explicit feature-sentiment pairs. Rana et al. (2017) propose a two-fold rules-based model, using rules defined by sequential patterns. Their first fold extracts aspects associated with domain independent opinions and the second fold extracts aspects associated with domain dependent opinions.

However, such rule-based models are designed for extracting product features which can not easily adapt to our K most prominent aspect extraction problem. Besides, most of them require human efforts to collect lexicons and to carefully design complex rules and thus do not scale very well.

4.2 Topic Modeling Based Methods

Most work in this domain are based on two basic models, pLSA(Hofmann, 1999) and LDA(Blei et al., 2003). The variants of these models consider two special features of review texts: 1) topics shift quickly between sentences, 2) sentiment plays an important role and there is a strong correlation between sentiments and aspects. The approach of Lin et al. (2011) models are parallel aspects and sentiments per review. Lin et al. (2009) models the dependency between the latent aspects and ratings. Wang et al. (2011a) proposed a generative model which incorporates topic modeling technique into the latent rating regression model (Wang et al., 2010). Moghaddam et al. (2012) made a nice summarization of some basic variations of LDA for opinion mining. In stead of using topics, our method relies on word embeddings to capture the latent semantics of words and phrases and achieves better results. *MG-LDA* (Titov and McDonald, 2008) is a variant of LDA that can also model topics at different granularities, which are based on extensions to standard topic modeling methods such as LDA and PLSA to induce multigrain topics. *D-PLDA* (Moghaddam and Ester, 2012), is a variant of LDA models, which is designed specifically for modeling topics from user reviews. D-PLDA only considers opinion-related terms and phrases, and nouns and phrases are controlled by two separate hidden parameters. Thus, the model needs aspects, ratings, and phrases as input, which are all very expensive.

4.3 Neural Network Based Methods

He et al. (2017) propose a neural attention model for identifying aspect terms. Their goal is similar to ours but instead of directly comparing their extracted terms with the gold standard, they ask human judges to map the extracted terms to one of the prominent gold aspects manually before computing the precision/recall. This evaluation methodology mixed machine results with human judgment and is problematic in our opinion. Our experiments showed that their output aspects are too fine-grained and can not be used as prominent aspects.

5 Conclusion

In this paper, we propose an unsupervised framework ExtRA for extracting the most prominent aspect terms about a type of product or service from user reviews, which benefits both qualitative and quantitative aspect-based review summarization. Using WordNet as a backbone, and by running personalized page rank on the network, we can produce aspect terms that are both important and non-overlapping. Results show that this approach is more effective than a number of other strong baselines.

Acknowledgment

Kenny Q. Zhu is the contact author and was supported by NSFC grants 91646205 and 61373031. Thanks to the anonymous reviewers for their valuable feedback.

References

Satanjeev Banerjee and Ted Pedersen. 2003. Extended gloss overlaps as a measure of semantic relatedness. In *Ijcai*, volume 3, pages 805–810.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *Journal of Machine Learning Research*.

Samuel Brody and Noemie Elhadad. 2010. An unsupervised aspect-sentiment model for online reviews. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 804–812. Association for Computational Linguistics.

Xueqi Cheng, Xiaohui Yan, Yanyan Lan, and Jiafeng Guo. 2014. Btm: Topic modeling over short texts. *TKDE*.

Xiaowen Ding, Bing Liu, and Philip S Yu. 2008. A holistic lexicon-based approach to opinion mining. In *Proc. WSDM*.

Gayatree Ganu, Noemie Elhadad, and Amélie Marian. 2009. Beyond the stars: improving rating predictions using review text content. In *WebDB*, volume 9, pages 1–6. Citeseer.

Stefan Gindl, Albert Weichselbraun, and Arno Scharl. 2013. Rule-based opinion target and aspect extraction to acquire affective knowledge. In *Proc. of WWW*.

Ruidan He, Wee Sun Lee, Hwee Tou Ng, and Daniel Dahlmeier. 2017. An unsupervised neural attention model for aspect extraction. In *ACL*.

Thomas Hofmann. 1999. Probabilistic latent semantic indexing. In *Proc. of SIGIR*.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proc. of KDD*.

L Kou, George Markowsky, and Leonard Berman. 1981. A fast algorithm for steiner trees. *Acta informatica*.

Himabindu Lakkaraju, Chiranjib Bhattacharyya, Indrajit Bhattacharya, and Srujana Merugu. 2011. Exploiting coherence for the simultaneous discovery of latent facets and associated sentiments. In *SDM*.

Chenghua Lin and Yulan He. 2009. Joint sentiment/topic model for sentiment analysis. In *Proc. of CIKM*.

Jialu Liu, Jingbo Shang, and Jiawei Han. 2017. Phrase mining from massive text and its applications. *Synthesis Lectures on Data Mining and Knowledge Discovery*.

Qian Liu, Zhiqiang Gao, Bing Liu, and Yuanlin Zhang. 2015. Automated rule selection for aspect extraction in opinion mining. In *IJCAI*, volume 15, pages 1291–1297.

Qian Liu, Bing Liu, Yuanlin Zhang, Doo Soon Kim, and Zhiqiang Gao. 2016. Improving opinion aspect extraction using semantic similarity and aspect associations.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Proc. of NIPS*.

Samaneh Moghaddam and Martin Ester. 2012. On the design of lda models for aspect-based opinion mining. In *Proc. of CIKM*.

John Pavlopoulos and Ion Androutsopoulos. 2014. Aspect term extraction for sentiment analysis: New datasets, new evaluation measures and an improved unsupervised method. *Proceedings of LASMEACL*.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proc. of EMNLP*.

Ana-Maria Popescu and Orena Etzioni. 2007. Extracting product features and opinions from reviews. In *Natural language processing and text mining*.

Soujanya Poria, Erik Cambria, Lun-Wei Ku, Chen Gui, and Alexander Gelbukh. 2014. A rule-based approach to aspect extraction from product reviews. In *Proc. of the Workshop on Natural Language Processing for Social Media*.

Guang Qiu, Bing Liu, Jiajun Bu, and Chun Chen. 2011. Opinion word expansion and target extraction through double propagation. *CL*.

Toqir A Rana and Yu-N Cheah. 2017. A two-fold rule-based model for aspect extraction. *Expert Systems with Applications*.

Peter J Rousseeuw. 1987. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, 20:53–65.

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 *Proc. of WWW*.

Liling Tan. 2014. Pywsd: Python implementations of word sense disambiguation (wsd) technologies [software].

Ivan Titov and Ryan McDonald. 2008. Modeling online reviews with multi-grain topic models. In *WWW*.

Hongning Wang, Yue Lu, and Chengxiang Zhai. 2010. Latent aspect rating analysis on review text data: a rating regression approach. In *Proc. of KDD*.

Hongning Wang, Yue Lu, and ChengXiang Zhai. 2011a. Latent aspect rating analysis without aspect keyword supervision. In *Proc. of KDD*.

Hongning Wang, Chi Wang, ChengXiang Zhai, and Jiawei Han. 2011b. Learning online discussion structures by conditional random fields. In *SIGIR*.

Linlin Wang, Kang Liu, Zhu Cao, Jun Zhao, and Gerard de Melo. 2015. Sentiment-aspect extraction based on restricted boltzmann machines. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, volume 1, pages 616–625.

Lingwei Zeng and Fang Li. 2013. A classification-based approach for implicit feature identification. In *Chinese Computational Linguistics and Natural Language Processing Based on Naturally Annotated Big Data*.

Wayne Xin Zhao, Jing Jiang, Hongfei Yan, and Xiaoming Li. 2010. 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. Association for Computational Linguistics.