Knowledge empowered prominent aspect extraction from product reviews

Zhiyi Luo, Shanshan Huang, Kenny Q. Zhu*

[a]*Department of Computer Science and Engineering, Shanghai Jiao Tong University*
[b] *800 Dongchuan Road, Shanghai, China 200240*

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 method empowered by knowledge sources such as Probase and WordNet, for extracting the most prominent aspects of a given product type from textual reviews. The proposed method, ExtRA (Extraction of Prominent Review Aspects), ~~infers~~ (i) extracts the aspect candidates from text reviews based on a data driven approach, (ii)builds an aspect graph utilizing the Probase to narrow the aspect space, then (iii) separates the space into reasonable aspect clusters ~~and extracts~~ by employing a set of proposed algorithms and finally (iv) generates K most prominent aspect terms or phrases which do not overlap semantically automatically without supervision from those aspect clusters. It is general-purpose and can be applied to almost any type of product and service. Extensive experiments show that ExtRA is effective and achieves the state-of-the-art performance on a dataset consisting of different product types.

Keywords: Prominent aspect extraction; Unsupervised learning; Topic modeling; Word embedding

---

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

*Corresponding author
Email addresses: jessherlock@sjtu.edu.cn (Zhiyi Luo), huangss_33@sjtu.edu.cn (Shanshan Huang), kzhu@cs.sjtu.edu.cn (Kenny Q. Zhu)

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 rate the hotel on several specific aspects of, 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.

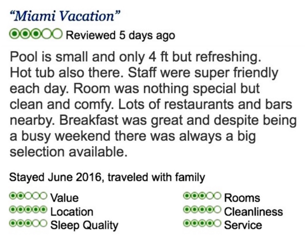
Existing approaches for producing such prominent aspect terms have been largely manual work [1, 2]. This is feasible for web services that only sell (or review) a small number of product types of the same domain. For example, TripAdvisor.com only features travel-related products, and Cars.com only reviews 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* ~~[3, 4]~~[3, 4, 5]. 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 (such as "appearance") 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. Besides, this task, as a downstream application, also benefits from the studies on fine-grained aspect extraction [6, 7]. However, such fine-grained aspects are not suitable for the applications such as aspect-based review summary.

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

- The extracted prominent aspects not only need to cover as many customer


"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.

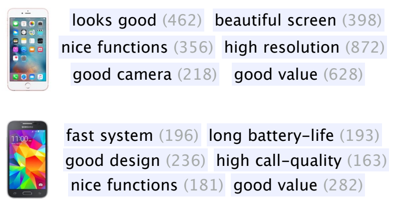


Figure 2: Automatic review summarization for two mobile phones on an e-commerce website

concerns as possible but also have little semantic overlap.

- The expression of user opinions is highly versatile: aspect terms can be expressed by different ways. For example, the mention of "cost" implies the aspect "price".
- 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 [8, 9, 10]. The main problem of such approaches is that they typically use only word frequency and co-occurrence information, and thus the performance degrades 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 approach extracts K most prominent aspect terms in three main steps: first it extracts potential aspect terms from text corpus by lexico-syntactic analysis; then we proposed a novel method to aggregate the extracted terms into *aspect clusters* by utilizing the external knowledge source, including WordNet [11] and Probase [12]; after that it ranks the *aspect clusters* by a proposed importance score; and finally we design an algorithm to generate the K prominent aspects from the aspect synonyms.

The main contributions in this paper are as follows:

1. We define a new problem, extracting prominent aspects from customer review corpora, which benefits existing aspect-based review analysis systems.
2. We propose a novel unsupervised and effective method ExtRA which utilize Probase and WordNet as well as word embeddings for inferring reasonable aspect clusters and extracting prominent aspects. Extensive experiments show that our approach outperforms multiple strong baselines by a substantial margin (Section 5).
3. We will release the implementation of the system and the evaluation dataset for future work in this research area.

2. Related Work

Existing research on *aspect-based review analysis* has focused on mining opinion based on given aspects ~~[3, 4]~~ [3, 4, 13, 14, 15] or jointly extracting the aspects and sentiment ~~[9, 16? , 17, 18]~~[9, 16, 17, 18, 19, 20]. 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.

2.1. Rule-based Methods

These methods leverage word statistical and syntactic features to manually design rules, recognizing aspect candidates from texts. Poria et al. [1] use manually crafted mining rules. Qiu et al. [**?**] also used rules, plus the Double Propagation method to better relate sentiment to aspects. Gindl et al. [21] cooperate the Double Propagation with anaphora resolution for identifying co-references to improve the accuracy. Su et al. [3] 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. [4] mapped implicit features to explicit features using a set of sentiment words and by clustering explicit feature-sentiment pairs. Rana et al. [22] 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.

2.2. Topic Modeling Based Methods

Most work in this domain are based on two basic models, pLSA[23] and LDA[24]. 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. Shams [25] incorporates co-occurrence relations as prior domain knowledge into LDA model. The approach of Lin et al. [8] models ~~are~~ parallel aspects and sentiments per review. Lin et al.~~[9] models~~ [9] and Moghaddam et al. [26] model the dependency between the latent aspects and ratings. Wang et al. [10] proposed a generative model which incorporates topic modeling technique into the latent rating regression model [27]. Moghaddam et al. [28] 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* [29] 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 multi-grain topics. *D-PLDA* [28], is a variant of LDA models is designed specifically for modeling topics from user reviews. D-PLDA utilizes the overall ratings of reviews for jointly mining the aspects and opinions for a given product. However, collecting such ratings for product reviews is often expensive. Thus, D-PLDA is difficult to scale to and apply to the large and cross-domain services.

2.3. Neural Network Based Methods

Several works [14**?**] based on the deep learning approach targets at the fine-grained aspect extraction. And most recently, He et al. [30] 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.

3. ~~Approach~~Problem Statement

~~depicts the architecture of ExtRA. The collection of customer reviews and the number of prominent aspects to extract are inputs. ExtRA builds aspect graph and infers aspect clusters by relation weighing and grouping, and then the top K prominent aspects are generated. ExtRA is makes use of external knowledge such as Probase, WordNet and word embeddings.~~

~~The architecture of ExtRA~~

3.1. ~~Problem Statement~~

~~The review aspect extraction~~ As mentioned before, our focus in this work is to generate most prominent aspects of a given product type from textual reviews. More specifically, the problem is defined as: 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*. Here K is an constant parameter for the problem. The set of reviews and the number of aspects are inputs.

Definition 1. We term the problem as *Prominent Aspect Extraction*, which aims to find an optimal set of K aspects $[a_1, a_2, ..., a_K]$, such that:

- the K aspects cover as much aspect space C as possible, i.e. maximizing $\bigcup_{k=1}^{K} c(a_k)$;
- the semantic overlaps between those aspects are minimum, i.e. minimizing $\sum_{i=1}^{K} \sum_{j=1}^{K} o(a_i, a_j)$,

where $c(a_k)$ denotes the semantic coverage of the aspect a_k, and $o(a_i, a_j)$ represents the semantic overlaps between a_i and a_j.

The challenges in a good choice of $a_1 \sim a_K$ are properly formulating the *semantic coverage* and *semantic overlaps*. To formulate the aspect space with minimum noise, ExtRA builds an aspect graph specialized for the given type of product. Then, we propose a novel strategy which segments the space into reasonable aspect clusters in order to model the semantic coverage of aspects more accurate, since the potential prominent aspects can benefit from other members within the same cluster. Finally, we generate the prominent aspects from aspect clusters by utilizing the internal structure of each aspect cluster.

Note that in this definition we don't use cross-domain information, that is, for one product type we only use the reviews of that domain. Thus, our model can extract aspects from any domain as long as the reviews for that domain is available.

4. Methodology

Figure 3 depicts the architecture of ExtRA. The collection of customer reviews and the number of prominent aspects to extract are inputs. ExtRA extracts the aspect candidates from text reviews, and builds aspect graph to narrow the aspect space, then segments the space into aspect clusters utilizing strategies of relation weighting and grouping, and finally generates the top K prominent aspects from aspect clusters. ExtRA is makes use of external knowledge such as Probase, WordNet and word embeddings.

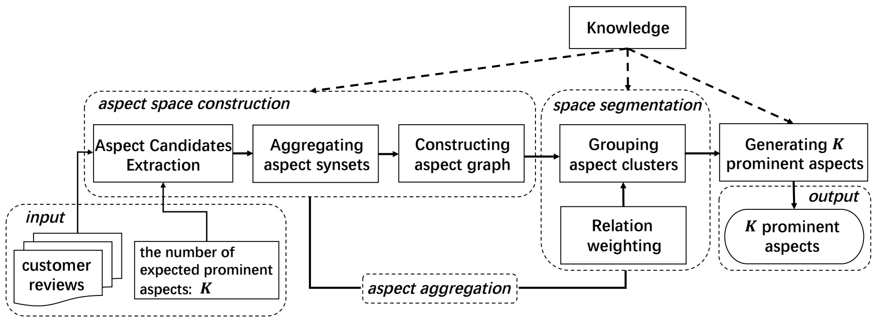

Knowledge
aspect space construction
space segmentation
Aspect Candidates Extraction
Aggregating aspect synsets
Constructing aspect graph
Grouping aspect clusters
Generating K prominent aspects
Relation weighting
input
customer reviews
the number of expected prominent aspects: K
aspect aggregation
output
K prominent aspects

Figure 3: The architecture of ExtRA

4.1. Aspect Candidates Extraction

Following the observation of Liu [31, 32], 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. 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 introduce *noun compound* relation (abbreviated *comp*) to extend phrases as aspect candidates. The phrase extension rules for the extracted aspect-opinion pair $\langle N, A\rangle$ are as follows:

Rule E1. If $\langle N, A\rangle$ and $comp(N_{-1}, N)$, then use $\langle N_{-1}\_N, A\rangle$ to replace $\langle N, A\rangle$;

Rule E2. If $\langle N, A\rangle$ and $comp(N, N_{+1})$, then use $\langle N\_N_{+1}, A\rangle$ to replace $\langle N, A\rangle$;

Rule E3. If $\langle N, A\rangle$ and $amod(V_{-1}, N)$, then use $\langle V_{-1}\_N, A\rangle$ to replace $\langle N, A\rangle$,

where N_{-1} and N_{+1} denotes the noun word, V_{-1} is the gerund, and the subscript represents the displacement to N in the sentence. Note that in order to obtain more accurate aspect candidates, we aim to extract the sentiment aspect-opinion pairs. We constrain that the opinion word A of each pair $\langle$N, A$\rangle$ is in the sentiment opinion lexicon proposed by Liu[31].

Figure 4 demonstrates the extraction process using some example sentences with syntactic features. For example, we extract the pair $\langle$*great*, *zoom_lens*$\rangle$ from sentence (a) by applying *Rules 1* and *E1*. Similarly, the extraction rules match $\langle$*slow*, *shutter*$\rangle$, $\langle$*noisy*, *shutter*$\rangle$ in sentence (b) and $\langle$*long*, *charging_time*$\rangle$ in sentence (c) as potential aspect-opinion pairs.

After extracting such aspect-opinion pairs from the text reviews, we use $freq(\langle N, A\rangle)$ to represent the number of occurrences of $\langle N, A\rangle$. The frequency of the aspect N (i.e. $freq(N)$) is computed as:

$$freq(N) = \sum_{\langle N,A\rangle \in P} freq(\langle N, A\rangle), \tag{1}$$

where P is the set of aspect-opinion pairs extracted from the corpus. We sort the extracted aspects by frequency, and consider the top 1000 of them as aspect candidates, assuming that the most prominent aspects are subsumed by those candidates terms.

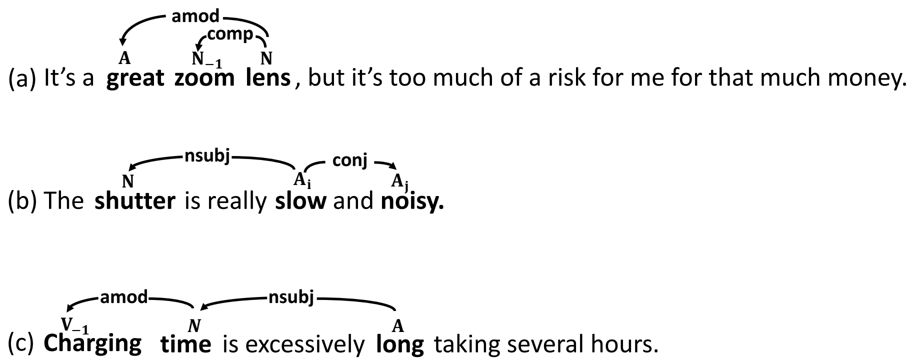

amod
comp
A
N_{-1}
N
(a) It's a great zoom lens, but it's too much of a risk for me for that much money.
nsubj
conj
N
A_i
A_j
(b) The shutter is really slow and noisy.
amod
nsubj
V_{-1}
N
A
(c) Charging time is excessively long taking several hours.

Figure 4: The example sentences for illustrating the aspect-opinion pairs extraction

4.2. Aspect Aggregation

We collect a bunch of aspect candidates from the previous extraction stage. The *prominent aspects* are supposed to be important and popular but sometimes coarse grained, which means they may semantically cover or overlap with a set of other aspect candidates. The simplest method is to extract the most frequent K aspect candidates as the K expected prominent aspects since the frequency is a natural measure of popularity (or importance). However, the expression of the prominent aspect can be highly versatile. For example, *os*, *operating system* and *Windows XP* describe the same prominent aspect of Laptop: *operating system*. Therefore, it is reasonable to aggregate the aspects which are about the same prominent aspect together to enhance the popularity. We aggregate the aspect candidates into clusters following two principles:

- ~~Aggregating~~ Aggregate the aspects with high semantic overlaps into *aspect synsets*.
- ~~Grouping~~ Group the *aspect synsets* with high semantic ~~coverage~~ subsumption into *aspect clusters*.

~~We use several external knowledge sources including WordNet[11], Glove[33] and Probase[12] to capture such semantic overlap and coverage between the extracted aspect candidates.~~ In this section, we first aggregate the aspect candidates into aspect synsets in Section 4.2.1, ensuring that the candidates which refer to the same aspect are attached with each other. Then, we narrow the aspect space from the original set of aspect candidates to the set of nodes (aspects) of the constructed aspect graph in Section 4.2.2. Finally, we utilize the proposed relation weighting scheme to segment the aspect space into clusters.

4.2.1. Aspect synsets

We use WordNet and Glove to aggregate the aspect candidates with highly semantic overlaps into *aspect synsets*. *Aspect synonyms* are the aspects that have similar meanings. *Aspect synset* is a group of aspect synonyms. The algorithm of aspect synsets generation is described in Algorithm 1.

Algorithm 1: Aspect Synsets Generation

Input: A list of aspects $N = [n_1, n_2, ..., n_M]$ sorted by frequency in descending order.

Output: A list of aspect synsets $S = [s_1, s_2, ..., s_T]$ sorted by priority in descending order.

```
1  $S \leftarrow []$;
2  $toSkip \leftarrow \emptyset$;
3  for $i \leftarrow 1$ to $M$ do
4      if $n_i \in toSkip$ then
5          continue;
6      $t \leftarrow len(S)$;
7      $s_t \leftarrow \{n_i\}$;
8      $L_i \leftarrow lemmas(n_i)$;
9      for $j \leftarrow i + 1$ to $M$ do
10         if $n_j \in toSkip$ then
11             continue;
12         $L_j \leftarrow lemmas(n_j)$;
13         if $n_i \in L_j$ and $n_j \in L_i$ or $sim(n_i, n_j) \geq thred$ then
14             $s_t \leftarrow s_t \cup \{n_j\}$;
15             $toSkip \leftarrow toSkip \cup \{n_j\}$;
16         for $n \in L_j$ do
17             if $n \in N$ and $n_j \in lemmas(n)$ or $sim(n, n_j) \geq thred$ then
18                 $s_t \leftarrow s_t \cup \{n\}$;
19                 $toSkip \leftarrow toSkip \cup \{n\}$;
20     Append $s_t$ to $S$;
21     $toSkip \leftarrow toSkip \cup \{n_i\}$;
22 Sort $S$ by priority in descending order;
23 return $S$;
```

WordNet. The WordNet organizes words with similar meanings as *synsets*. A *synset* represents a specific sense of a specific word. Each synset contains one or more lemmas, vice versa, one lemma can belong to one or more synsets. Thus, it is intuitive to take the aspects that belong to the same *synset* as the aspect synonyms. Such aspect synonyms are supposed to have the same meaning at least in one sense. We ensure that each aspect (i.e. n) is assigned to only one *aspect synset* using Algorithm 1. $lemmas(n)$ represents the set of all synonyms for n. For example, word n have m senses which corresponds to one synset each in WordNet. $l(s)$ is the set of synonyms of synset s. Then, $lemmas(n) = \bigcup_{i=1}^{m} l(s_i)$. Note that the priority of each *aspect synset* (e.g. s) is computed as:

$$\sum_{n \in s} freq(n). \tag{2}$$

Such aggregation groups *os* and *operating system* together.

Glove. While WordNet is powerful, the synsets information for phrases in WordNet is not as rich as words. For example, we expect to group *pool* and *swimming pool* which are almost the same meaning in semantics together, though they do not have the same sense in WordNet. To ameliorate this limitation, we introduce word embeddings which are widely used for capturing the semantic similarity (i.e. semantic overlap) between words and phrases. We calculate the semantic overlap between term n_i and n_j as $sim(n_i, n_j)$, which is the cosine similarity computed by Glove embeddings. If n_i is a phrase composed by words $[w_1, ..., w_K]$, then $E(n_i)$, the embedding of n_i, is calculated as:

$$E(n_i) = \sum_{k=1}^{K} E(w_k). \tag{3}$$

We set the threshold *thred* to be very high (i.e. 0.9) in order to group aspect synonyms in Algorithm 1.

4.2.2. Aspect Clusters

In this section, we aggregate the aspect synsets into aspect clusters. Each aspect cluster implies a prominent aspect for the given product or service. We expect to capture the hierarchical structure between aspects and to group those hypernym-hyponym aspects together. For example, *Windows XP* is the hyponym aspect of *operating system* which means *Windows XP* is a kind of *operating system.*

Probase. We leverage Probase to identity *isA* relations between aspects. Probase is a data driven semantic network that consists of millions of fine-grained concepts and their *isA* relationships. Thus, it is more suitable than WordNet on measuring the semantic subsumptions beween aspects. For an *isA* relationship $\langle u, v \rangle$ from Probase, the hypernym u is called *concept*, and the hyponym v is called *entity* or *instance.* Each relationship $\langle u, v \rangle$ is associated with the number of occurrences (i.e. $f(\langle u, v \rangle)$) that u occurred as the concept of v in the corpus. If $f(\langle u, v \rangle)$ is greater than 100, we consider that there is an isA relationship between u and v. In another word, v is semantically covered by u in the commonsense.

Aspect graph construction. Given the extracted aspect candidates, we first extract an aspect graph $G = (V, E)$ from Probase. Each node $v \in V$ is an aspect candidate. Each edge $e = \langle u, v \rangle$, directed from the concept u to the entity e, represents an isA relationship from Probase. To improve the quality of relationships in G, we only reserve top frequent relationships for both u and v. The algorithm for the graph construction is described in Algorithm 2. *topEntities*(n) returns the top 20 frequent entities for the concept n, and *topConcepts*(n) returns the top 20 frequent concepts for the entity n.

Algorithm 2: Aspect Graph Construction

```
    Input: The set of aspect candidates $N = \{n_1, n_2, ..., n_M\}$.
    Output: $G = (V, E)$.
 1  $V \leftarrow \emptyset$;
 2  $E \leftarrow \emptyset$;
 3  for $i \leftarrow 1$ to $M$ do
 4      for $j \leftarrow i + 1$ to $M$ do
 5          if $n_j \in topEntities(n_i)$ and $n_i \in topConcepts(n_j)$ then
 6              $E \leftarrow E \cup \{\langle n_i, n_j \rangle\}$;
 7          if $n_i \in topEntities(n_j)$ and $n_j \in topConcepts(n_i)$ then
 8              $E \leftarrow E \cup \{\langle n_j, n_i \rangle\}$;
 9  for $\langle u, v \rangle \in E$ do
10      $V \leftarrow V \cup \{u\}$;
11      $V \leftarrow V \cup \{v\}$;
12  return G ;
```

Relation weighting. In order to measure the quality of each relationship $\langle u, v \rangle$ in G, we propose a weighting scheme for $\langle u, v \rangle$. According to Wu [12], each relationship $\langle u, v \rangle$ is associated with two typicalities, $\mathcal{T}(u|v)$ and $\mathcal{T}(v|u)$, which are useful for conceptualization and inference. $\mathcal{T}(u|v)$ is the typicality of the concept u given the instance v, and $\mathcal{T}(v|u)$ is the typicality of the instance v given the concept u. In our scenario, they are computed as follows:

$$\mathcal{T}(u|v) = \frac{f(\langle u, v \rangle)}{f(v)} \tag{4}$$

$$\mathcal{T}(v|u) = \frac{f(\langle u, v \rangle)}{f(u)}, \tag{5}$$

where $f(u)$ (or $f(v)$) is the frequency of u (or v) acting as the concept (or instance) in Probase. The weight of the edge $\langle u, v \rangle$ is computed as:

$$w(\langle u, v \rangle) = \mathcal{T}(u|v)^{\frac{1}{2}} * \mathcal{T}(v|u)^{\frac{1}{2}} \tag{6}$$

This scheme captures the intuition that if u associates more typically to v or vice versa, $\langle u, v \rangle$ is often weighted higher.

Grouping. If $\langle u, v \rangle \in E$ and $\langle v, u \rangle \in E$, in another word, u is an instance of v and vice versa, then we take u and v as aspect synonyms. Thus, if $u \in s_u$ and $v \in s_v$, then we group the aspect synsets s_u and s_v together. In addition, we calculate the semantic similarity (i.e. *sim*) between the aspects and group the aspect synsets with top most similar aspects together. This step can be treated as the further enrichment of the aspect synsets.

Next, we group the aspect synsets $S = [s_1, s_2, ..., s_T]$ into aspect clusters $C = [c_1, c_2, ..., c_Q]$. The intuition is that if s is semantically covered by s',

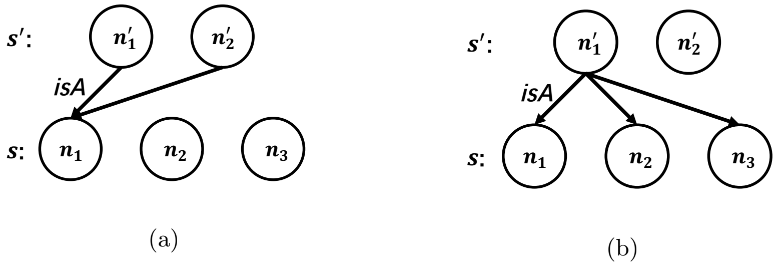


Figure 5: Two structures that s' covers s

which means they support the same prominent aspect, then we group s and s' together. Here, s (e.g. $s = \{n_1, n_2, n_3\}$) and s' (e.g. $s' = \{n'_1, n'_2\}$) are aspect synsets. We assume that the two structures shown in Figure 5 imply the semantical subsumption . Formally, structure (a) represents that $\exists n \in s$ such that $\langle n', n\rangle \in E$ for $\{\forall n', n' \in s'\}$. Similarly, structure (b) represents that $\exists n' \in s'$ such that $\langle n', n\rangle \in E$ for $\{\forall n, n \in s\}$. Either semantic coverage structure in Figure 5 should be satisfied when s is covered by s'.

The key of the grouping algorithm is to decide the grouping order for the aspect synsets. For example, s could be covered by multiple aspect synsets, such as s' and s'', It is too aggressive to group s, s' and s'' into one aspect cluster directly. We measure the semantic coverage using the *relation weighting* scheme proposed in previous step. Formally, the semantic coverage of s for s' is computed as:

$$cov(\langle s', s\rangle) = \max_{\langle u,v\rangle \in edges(\langle s', s\rangle)} w(\langle u, v\rangle) \tag{7}$$

$edges(\langle s', s\rangle)$ is the set of edges which are involved in the structures shown in Figure 5. $w(\langle u, v\rangle)$ is calculated as Eq. (6). Then, we sort the pairs of aspect synsets (e.g. $\langle s', s\rangle$) to be grouped together by their coverages $cov(\langle s', s\rangle)$ in descending order.

To control the purity of aspect clusters, we propose an iterative algorithm to group those pairs. We show the grouping details in Algorithm 3. The *grouping rate* parameter determines how many pairs to be grouped in each iteration. The output of each iteration is a list of aspect clusters feeding into the next iteration as inputs.

Given the list of aspect clusters C, $toGroup(C)$ gives the pairs of aspect clusters which satisfy the structures in Figure 5. Note that we set the *grouping rate* as 1 in our algorithm in oder to avoid aggressive grouping. After such grouping, each output aspect cluster (co_i) can be seen as a coarse-grained aspect cluster. To properly adjust the granularity and purity of co_i ($co_i \in C_{out}$), we further cluster the aspects in co_i. We represent each aspect n as a vector described in Eq. (3) and use the K-means clustering method to cluster them.

Algorithm 3: Grouping aspect clusters

```
   Input: A list of aspect synsets $S = [s_1, s_2, ..., s_T]$.
   Output: A list of aspect clustsers $C_{out} = [co_1, co_2, ..., co_Q]$
1  Initialize $C_{in} = [ci_1, ci_2, ..., ci_T]$, where $ci_t = s_t$;
2  while true do
3      $C_{out} \leftarrow C_{in}$;
4      $pairs \leftarrow toGroup(C_{in})$;
5      if $len(pairs) = 0$ then
6          break;
7      Sort $\langle s', s\rangle \in pairs$ by $cov(\langle s', s\rangle)$ (Eq. (7)) in descending order, such that
        $pairs = [\langle s'_1, s_1\rangle, ..., \langle s'_K, s_K\rangle]$;
8      $combined \leftarrow \emptyset$;
9      $C \leftarrow []$;
10     for $i \leftarrow 1$ to $grouping\_rate$ do
11         $s \leftarrow s'_i \cup s_i$;
12         $combined \leftarrow combined \cup \{s'\} \cup \{s\}$;
13         Append $s$ to $C$;
14     for $s \in C_{in}$ and $s \notin combined$ do
15         Append $s$ to $C$;
16     Sort $C$ by priority (Eq. (2)) in descending order;
17     $C_{in} \leftarrow C$;
18 return $C_{out}$;
```

The number of clusters Z is auto-tuned using silhouette score [34] [1].

For example, after clustering, the co_i splits into Z clusters represented as $co_i^{(1)}$, $co_i^{(2)}$, ..., and $co_i^{(Z)}$. We treat such aspect clusters $co_i^{(j)}$ as fine-grained aspect clusters. Each $co_i^{(j)}$ is associated with two priorities. One is $priority(co_i^{(j)})$, and the other is $priority(co_i)$ which is the priority of its corresponding coarse-grained aspect cluster co_i. We use $priority(co_i)$ as the first key and $priority(co_i^{(j)})$ as the second key to sort the fine-grained aspect clusters $co_i^{(j)}$ in descending order. The prominent aspects we expected are implied in those top fine-grained aspect clusters.

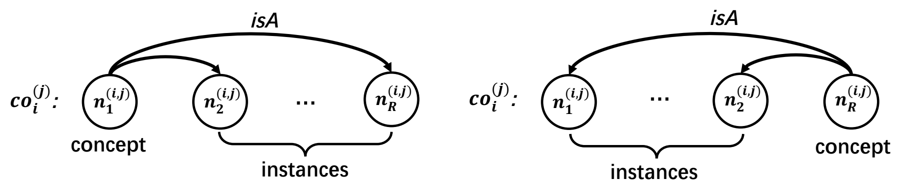
4.3. Prominent Aspects Generation

Finally, we generate the K prominent aspects from the top fine-grained aspect clusters $co_i^{(j)}$. We sort the aspects in $co_i^{(j)}$ by priority in descending order, representing as $[n_1^{(i,j)}, n_2^{(i,j)}, ..., n_R^{(i,j)}]$. Thus, $n_1^{(i,j)}$ is the most popular aspect with the highest frequency $freq(n_1^{(i,j)})$.

Next, we generate prominent aspects from $co_i^{(j)}$ on three different cases: *Top-down case*, *Bottom-up case* and *Miscellaneous case*. We demonstrate the

[1]We empirically set the default silhouette score as 0.13 for $Z = 1$.

first two cases in Figure 6.



(a) Top-down case (b) Bottom-up case

Figure 6: Different cases for prominent aspect generation from $co_i^{(j)}$

Top-down case. $n_1^{(i,j)}$ is the concept of $n_r^{(i,j)}(2 \leq r \leq R)$. In another word, $n_r^{(i,j)}(2 \leq r \leq R)$ are the instances of $n_1^{(i,j)}$. In this case, we extract the concept $n_1^{(i,j)}$ as the prominent aspect, except that $n_1^{(i,j)}$ is too vague (e.g. *feature, service*), as well as $n_1^{(i,j)}$ do not dominate $co_i^{(j)}$, then we extract the suboptimal aspect $n_2^{(i,j)}$ as the prominent aspect. We formulate the domination constraint of $n_1^{(i,j)}$ in $co_i^{(j)}$ as follows:

$$\frac{freq(n_1^{(i,j)})}{priority(co_i^{(j)})} \leq \tau^2 \tag{8}$$

Note that we measure the vagueness of each concept using Probase[12].

Bottom-up case. $n_R^{(i,j)}$ is the concept of $n_r^{(i,j)}(1 \leq r \leq R-1)$. In this case, we extract the most popular instance x and the concept $n_R^{(i,j)}$ as the prominent aspects. Formally,

$$x = \mathop{\arg\max}_{n_r^{(i,j)}, 1 \leq r \leq R-1} f(n_r^{(i,j)}) \tag{9}$$

Miscellaneous case. All the other cases are divided as the miscellaneous case. In this case, we simply extract the most popular aspect $n_1^{(i,j)}$ as the prominent aspect.

We perform above strategy along the sorted list of fine-grained aspect clusters until we generate K prominent aspects.

5. ~~Experiments~~Experimental Results

We compare ExtRA 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.

$^2\tau = \frac{3}{4}$

5.1. Dataset

We use the customer review corpora of 6 kinds of product and service [3] collected from popular websites, including Amazon, TripAdvisor and Yelp. The number of hotel reviews [35] 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 [31, 36, 37, 38] 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 [39, 40, 16, 17] as well as the popular commercial websites (e.g. TripAdvisor), which commonly label 4 to 9 prominent aspects for rating, we respectively set K as 5, 7 and 9 in the following experiments. We have five annotators in total. We ask each annotator who are familiar with the domain to give K aspect terms which they think are most important for each category. [4]

One prominent aspect could 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 5.3.1 to compensate this problem. To acquire a relatively higher inner-agreement, we educate the annotators with top 100 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 2 is computed as the average jaccard similarity between every two annotators. As can be seen, for the category *hotel*, the agreement between annotators achieves the highest value 0.44 when K is 5, which means that $K = 5$ is the most reasonable number of prominent aspects for *hotel*. Similarly, $K = 9$ is the most appropriate value for *mp3*, *cameras*, *mobile phone* and *laptop*. and it is better to set K as 7 for *restaurant*. As seen from the released evaluation dataset, although the jaccard agreements on *cameras* and *laptop* is relatively low, the collected labels are still highly correlated on semantics.

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

[4]The complete labeled set of ExtRA are released at `https://drive.google.com/file/d/1Ot0Y4YSUtW7cRQWgfaHvLvECjOaPRuWo/view?usp=sharing`.

Table 2: Inner-annotator agreements

Category / Agreement	hotel	mp3	cameras	mobile phone	laptop	restaurant
$K = 5$	**0.440**	0.258	0.201	0.148	0.131	0.464
$K = 7$	0.362	0.375	0.277	0.319	0.225	**0.497**
$K = 9$	0.365	**0.405**	**0.301**	**0.431**	**0.250**	0.456

5.2. Baselines and ExtRA

We introduce three topic modeling based baselines for the task. These are **LDA** [24], **BTM** ~~[?]~~[41] and **MG-LDA** [29]. 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, 7 and 9 separately. The hyper-parameter of MG-LDA (global topics) is set to 30 with fine-tuning.

Another syntactic rule-based baseline model **RuleExt** is from the first step of the proposed method. After extracting the aspect candidates using *rules* (i.e. Rule 1$\sim$ 3) and *extension rules* (i.e. Rule E1$\sim$ E3) in Section 4.1, we sort the aspect candidates by their counts of extracted occurrences. Then select the top K candidates as the prominent aspects.

ABAE [30] 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. We set the number of training epochs as 100 using the code [5] released by author.

Note that the top aspect terms from the baseline methods, inevitably contain some sentiment words and opinions, like *excellent, nice, great, etc.* To remedy this, we post process the results for baselines methods in order to reverse the nouns only.

For **ExtRA**, we use WordNet 3.0 and the Probase snapshot released in 2012 which consists of $69,707,026$ distinct concept-instance pairs in total. We use GloVe embeddings with 300 dimensions, trained from 840B tokens using common crawl data. As described in Section 4.2.2, we use silhouette score to automatically tune the cluster number Z when generating the final prominent aspect clusters for $co_i^{(j)}$ by K-means. We search the value of Z from 1 to $min(V-2, \frac{V}{2})$, where $V = |co_i^{(j)}|$. The optimal Z achieves the highest silhouette score after clustering.

[5]https://github.com/ruidan/Unsupervised-Aspect-Extraction

5.3. *Evaluation*

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

5.3.1. *Quantitative Evaluation*

410 We evaluate our model as well as above baselines on the evaluation dataset described above. We have $H = 5$ human annotators in total. We did not remove the duplicate aspect labels for the qualitative evaluation, since the repeated aspects are assume to be better.

Table 3: Comparison of *hard* (upper row) & *soft* (lower row) accuracies using different models with optimal K for each category.

	LDA	BTM	MG-LDA	ABAE	RuleExt	ExtRA
hotel	0.360	0.360	0.360	0.120	0.520	**0.560**
$(K = 5)$	0.613	0.625	0.614	0.394	0.704	**0.769**
mp3	0.089	0.089	0.178	0.000	0.222	**0.467**
$(K = 9)$	0.396	0.402	0.542	0.304	0.547	**0.689**
cameras	0.178	0.133	0.133	0.133	0.133	**0.422**
$(K = 9)$	0.509	0.485	0.490	0.441	0.507	**0.678**
mobile phone	0.200	0.200	0.222	0.000	0.333	**0.511**
$(K = 9)$	0.488	0.505	0.524	0.227	0.633	**0.701**
laptop	0.200	0.044	0.200	0.000	0.200	**0.311**
$(K = 9)$	0.491	0.364	0.504	0.261	0.516	**0.630**
restaurant	0.257	0.114	0.286	0.000	0.371	**0.600**
$(K = 7)$	0.548	0.403	0.560	0.232	0.627	**0.749**

5.3.2. ~~*Quantitative Evaluation*~~

~~We evaluate our model as well as above baselines on the evaluation dataset described above. We have $H = 5$ human annotators in total. 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 $H * K$ labels that exactly match one of the K aspect terms generated by the model as the *hard accuracy* of the model. Formally, $Aspects(m) = [a_1, a_2, ..., a_K]$ denotes the K prominent aspects generated from model m for the given category. $L = [l_1, l_2, ..., l_{H*K}]$ are the $H * K$ golden aspect terms, where $L^{(h)} = [l_{(h-1)*K+1}, ..., l_{h*K}]$ are from the h-th annotator. The hard accuracy is defined as:

$$hacc(m) = \frac{\sum_{i=1}^{H*K} hit(Aspects(m), l_i)}{25} \tag{10}$$

$$hit(Aspects(m), l_i) = \begin{cases} 1, & l_i \in Aspects(m) \\ 0, & \text{otherwise,} \end{cases} \tag{11}$$

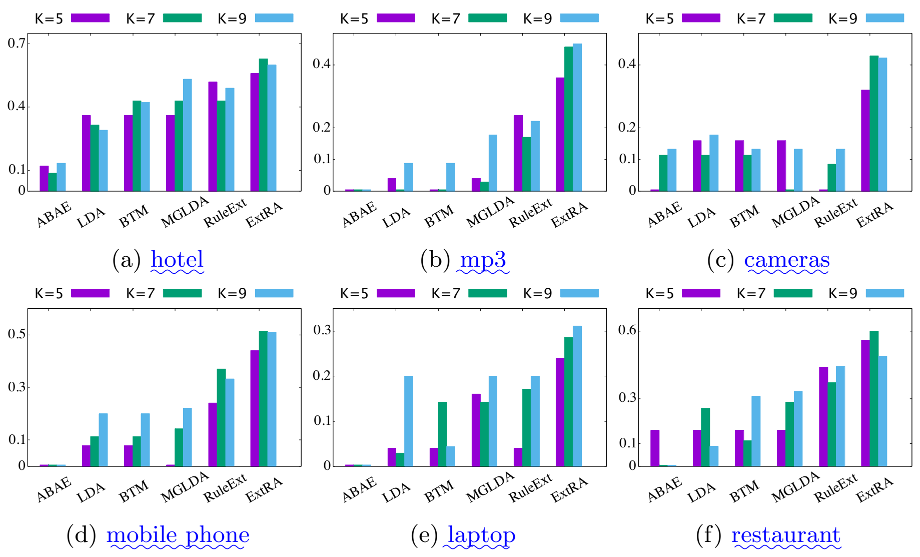


Figure 7: Comparison of hard accuracies on all Ks and categories

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.

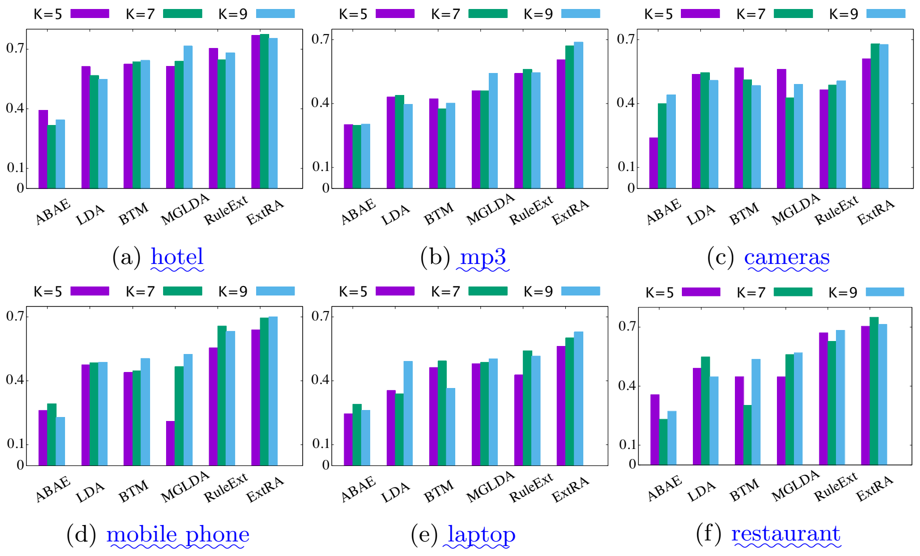


Figure 8: Comparison of soft accuracies on all Ks and categories

To remedy this, we propose the *soft accuracy* evaluation measure. 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 $align^{(h)}(a_k)$ achieves 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 [33]. We then compute the soft accuracy measure as follows:

$$sacc(m) = \frac{1}{H} \sum_{h=1}^{H} \sum_{k=1}^{K} sim(a_k, align^{(h)}(a_k)), \tag{12}$$

where K could be 5, 7 and 9 in the experiments. We show the hard and soft accuracies in Table 3 under the optimal K for each category. The complete comparison results including $K = 5, 7, 9$ for all categories are shown in Figure 7 (hard accuracy) and Figure 8 (soft accuracy). We can see that model ExtRA outperforms all the other baselines in all categories under both hard and soft accuracy measures.

5.3.2. Qualitative Analysis

To qualitatively evaluate different models, we present the extracted aspect terms by each model with the number of optimal K from each domain in Table 5 and Table 6.

Table 4: Comparison of aspect clusters for PAE and ABAE on *hotel*.

ABAE	**room**, bathroom, bed, bedroom, miniscule, ensuite
	accomodation, solamar, donatello, stay
	repeatedly, shouted, apologised, **supervisor**, yelled
	excellent, great, terrific, awesome, **divine**, sensational
	alexanderplatz, loop, **tunnel**, block
ExtRA	**room**, dining room, dining experience
	breakfast, breakfast buffet, buffet
	staff, reception staff, reception
	location, place, side, position
	price, cost
	bed, linen, furniture, chair, furnishings, bed size
	service, **internet**, internet access, shuttle service
	bathroom, shower, tub, bath
	pool, pool area, amenity, facility, swimming pool

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. battery life*. The proposed model avoid the overlapping aspects appeared in our strong baseline (RuleExt) by implicitly deduplication using our grouping algorithm. For example, both *picture* and *photo* are extracted as top aspects in the *cameras* category, but they mean nearly the same concept. In addition, ExtRA generate each prominent aspect associated with a group of supporting terms, expressing the prominent aspect in different ways while traditional baselines can only extract the prominent aspects. Among the baselines, only ABAE could infer such aspect clusters. We further show the effectiveness of ExtRA by comparing aspect clusters with ABAE in Table 4. For simplicity, we show the top terms in each aspect cluster on *hotel*. Each line in Figure 4 represents an aspect cluster, and we highlight the extracted prominent aspect from ExtRA with bold font. We can see that ABAE tends to cluster adjectives and verbs together which is not suitable for the prominent aspect extraction task.

Table 5: The prominent aspect terms for *hotel*, *mp3* and *cameras*

hotel ($K = 5$)	LDA	room, breakfast, location, pool, staff
	BTM	room, pool, location, staff, time
	MGLDA	room, breakfast, location, night, staff
	ABAE	room, accommodation, supervisor, divine, tunnel
	RuleExt	room, staff, hotel, location, bed
	ExtRA	**room, breakfast, staff, location, price**
mp3 ($K = 9$)	LDA	battery, button, work, player, music, video, ipod, quality, software
	BTM	battery, player, ipod, para, screen, work, music, software, headphone
	MGLDA	song, battery, ipod, player, computer, quality, button, software, size
	ABAE	album, tone, middle, dollar, application, birthday, lightweight, buyer, holder
	RuleExt	quality, sound, player, drive, screen, feature, battery, price, software
	ExtRA	**quality, sound, screen, control, display, headphone, battery life, price, software**
cameras ($K = 9$)	LDA	lens, battery, screen, canon, water, video, mode, problem, quality
	BTM	para, battery, light, canon, lens, video, mode, quality, card
	MGLDA	lens, manual, battery, light, canon, pocket, mode, year, quality
	ABAE	portability, battery, photo, vacation, illinois, lens, mode, device, promotion
	RuleExt	camera, *picture, photo*, quality, feature, shot, price, image, zoom
	ExtRA	**picture, zoom, lens, control, speed, price, battery life, accessory, screen**

6. Implications

Our main aim in this study was to address the problem of prominent aspect extraction, which benefits a multitude of downstream applications such as aspect-based

Table 6: The prominent aspect terms for *mobile phone*, *laptop* and *restaurant*

mobile phone ($K = 9$)	LDA	service, battery, screen, work, music, blackberry, android, price, button
	BTM	para, battery, screen, work, video, android, card, price, time
	MGLDA	battery, quality, screen, work, android, venezuela, card, text, price
	ABAE	flat, descent, kernel, bookmark, month, flagship, shutdown, representation, dealer
	RuleExt	phone, screen, camera, battery, price, quality, feature, picture, app
	ExtRA	**camera, quality, price, screen, battery life, accessory, speed, keyboard, reception**
laptop ($K = 9$)	LDA	google, apple, windows, screen, price, battery, game, problem, processor
	BTM	google, apple, windows, screen, game, video, problem, port, battery
	MGLDA	windows, screen, work, battery, speaker, keyboard, price, key, software
	ABAE	bridge, cable, twitter, bezel, window, shutdown, opinion, holiday, explanation
	RuleExt	laptop, computer, drive, screen, keyboard, battery, price, machine, feature
	ExtRA	**screen, graphic, price, drive, processor, hardware, battery life, system, speed**
restaurant ($K = 7$)	LDA	bar, room, service, food, time, pizza, chicken
	BTM	room, food, time, les, das, chicken, work
	MGLDA	star, service, area, food, time, chicken, minute
	ABAE	communication, killer, eatery, wright, chickpea, fireplace, tomorrow
	RuleExt	food, service, place, staff, price, experience, time
	ExtRA	**service, food, staff, flavor, atmosphere, selection, location**

review summarization. The existing methods mainly focus on fine-grained aspect extraction, which cannot adapt to the problem well.

We propose the ExtRA, an unsupervised and effective system, for generating aspect clusters as well as extracting prominent aspects from product reviews. In Section 5, we demonstrate the effectiveness of the components integrated within ExtRA. The key insight of extracting prominent aspects is that we value the aspect terms according to the popularity (or coverage) and the semantic overlaps with each other. We instantiate the popularity based on the data-driven approach of aspect extraction and utilize the knowledge sources to facilitate the specification of semantic overlaps between aspects. We further demonstrate the usefulness of the proposing strategies and especially that the relation weighting scheme, which is the point for the segmentation of aspect space, values the strength of subsumption relation between aspects based on the typicality score computed using Probase.

7. Conclusion

In this paper, we propose an knowledge empowered as well as unsupervised method 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. With the help of WordNet and Probase as knowledge source, and by utilizing the proposed algorithms, we can generate reasonable aspect clusters and produce the aspect terms that are both important and non-overlapping. Results show that this approach is more effective than a number of other strong baselines.

Acknowledgements

This work was supported by NSFC grant 61373031 and NSFC-NRF Joint Research Program No. 61411140247.

References

[1] S. Poria, E. Cambria, L.-W. Ku, C. Gui, A. Gelbukh, A rule-based approach to aspect extraction from product reviews, in: Proc. of the Workshop on Natural Language Processing for Social Media, 2014.

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

[3] Q. Su, X. Xu, H. Guo, Z. Guo, X. Wu, X. Zhang, B. Swen, Z. Su, Hidden sentiment association in chinese web opinion mining, in: Proc. of WWW, 2008.

[4] L. Zeng, F. Li, A classification-based approach for implicit feature identification, in: Chinese Computational Linguistics and Natural Language Processing Based on Naturally Annotated Big Data, 2013.

[5] A. S. Manek, P. D. Shenoy, M. C. Mohan, K. Venugopal, Aspect term extraction for sentiment analysis in large movie reviews using gini index feature selection method and svm classifier, World wide web 20 (2) (2017) 135–154.

[6] A.-D. Vo, Q.-P. Nguyen, C.-Y. Ock, Automatic knowledge extraction for aspect-based sentiment analysis of customer reviews, in: Proceedings of the 10th International Conference on Computer Modeling and Simulation, ACM, 2018, pp. 110–113.

[7] M. Dragoni, M. Federici, A. Rexha, An unsupervised aspect extraction strategy for monitoring real-time reviews stream, Information Processing & Management.

[8] H. Lakkaraju, C. Bhattacharyya, I. Bhattacharya, S. Merugu, Exploiting coherence for the simultaneous discovery of latent facets and associated sentiments, in: SDM, 2011.

[9] C. Lin, Y. He, Joint sentiment/topic model for sentiment analysis, in: Proc. of CIKM, 2009.

[10] H. Wang, Y. Lu, C. Zhai, Latent aspect rating analysis without aspect keyword supervision, in: Proc. of KDD, 2011.

[11] G. A. Miller, Wordnet: a lexical database for english, Communications of the ACM 38 (11) (1995) 39–41.

[12] W. Wu, H. Li, H. Wang, K. Q. Zhu, Probase: A probabilistic taxonomy for text understanding, in: Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data, ACM, 2012, pp. 481–492.

[13] T. A. Rana, Y.-N. Cheah, Improving aspect extraction using aspect frequency and semantic similarity-based approach for aspect-based sentiment analysis, in: International Conference on Computing and Information Technology, Springer, 2017, pp. 317–326.

[14] S. Poria, E. Cambria, A. Gelbukh, Aspect extraction for opinion mining with a deep convolutional neural network, Knowledge-Based Systems 108 (2016) 42–49.

[15] M. Tubishat, N. Idris, M. A. Abushariah, Implicit aspect extraction in sentiment analysis: Review, taxonomy, oppportunities, and open challenges, Information Processing & Management 54 (4) (2018) 545–563.

[16] W. X. Zhao, J. Jiang, H. Yan, X. Li, Jointly modeling aspects and opinions with a maxent-lda hybrid, in: Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, 2010, pp. 56–65.

[17] L. Wang, K. Liu, Z. Cao, J. Zhao, G. de Melo, 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), Vol. 1, 2015, pp. 616–625.

[18] Q. Liu, B. Liu, Y. Zhang, D. S. Kim, Z. Gao, Improving opinion aspect extraction using semantic similarity and aspect associations., 2016.

[19] C. Wu, F. Wu, S. Wu, Z. Yuan, Y. Huang, A hybrid unsupervised method for aspect term and opinion target extraction, Knowledge-Based Systems 148 (2018) 66–73.

[20] W. Wang, S. J. Pan, D. Dahlmeier, X. Xiao, Coupled multi-layer attentions for co-extraction of aspect and opinion terms., in: AAAI, 2017, pp. 3316–3322.

[21] S. Gindl, A. Weichselbraun, A. Scharl, Rule-based opinion target and aspect extraction to acquire affective knowledge, in: Proc. of WWW, 2013.

[22] T. A. Rana, Y.-N. Cheah, A two-fold rule-based model for aspect extraction, Expert Systems with Applications.

[23] T. Hofmann, Probabilistic latent semantic indexing, in: Proc. of SIGIR, 1999.

[24] D. M. Blei, A. Y. Ng, M. I. Jordan, Latent dirichlet allocation, Journal of Machine Learning Research.

[25] M. Shams, A. Baraani-Dastjerdi, Enriched lda (elda): Combination of latent dirichlet allocation with word co-occurrence analysis for aspect extraction, Expert Systems with Applications 80 (2017) 136–146.

[26] S. Moghaddam, M. Ester, Ilda: interdependent lda model for learning latent aspects and their ratings from online product reviews, in: Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval, ACM, 2011, pp. 665–674.

[27] H. Wang, Y. Lu, C. Zhai, Latent aspect rating analysis on review text data: a rating regression approach, in: Proc. of KDD, 2010.

[28] S. Moghaddam, M. Ester, On the design of lda models for aspect-based opinion mining, in: Proc. of CIKM, 2012.

[29] I. Titov, R. McDonald, Modeling online reviews with multi-grain topic models, in: WWW, 2008.

[30] R. He, W. S. Lee, H. T. Ng, D. Dahlmeier, An unsupervised neural attention model for aspect extraction, in: ACL, 2017.

[31] M. Hu, B. Liu, Mining and summarizing customer reviews, in: Proc. of KDD, 2004.

[32] Q. Liu, Z. Gao, B. Liu, Y. Zhang, Automated rule selection for aspect extraction in opinion mining., in: IJCAI, Vol. 15, 2015, pp. 1291–1297.

[33] J. Pennington, R. Socher, C. D. Manning, Glove: Global vectors for word representation, in: Proc. of EMNLP, 2014.

[34] P. J. Rousseeuw, Silhouettes: a graphical aid to the interpretation and validation of cluster analysis, Journal of computational and applied mathematics 20 (1987) 53–65.

[35] H. Wang, C. Wang, C. Zhai, J. Han, Learning online discussion structures by conditional random fields, in: SIGIR, 2011.

[36] A.-M. Popescu, O. Etzioni, Extracting product features and opinions from reviews, in: Natural language processing and text mining, 2007.

[37] J. Pavlopoulos, I. Androutsopoulos, Aspect term extraction for sentiment analysis: New datasets, new evaluation measures and an improved unsupervised method, Proceedings of LASMEACL.

[38] X. Ding, B. Liu, P. S. Yu, A holistic lexicon-based approach to opinion mining, in: Proc. WSDM, 2008.

[39] G. Ganu, N. Elhadad, A. Marian, Beyond the stars: improving rating predictions using review text content., in: WebDB, Vol. 9, Citeseer, 2009, pp. 1–6.

[40] S. Brody, N. Elhadad, 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, Association for Computational Linguistics, 2010, pp. 804–812.

[41] X. Cheng, X. Yan, Y. Lan, J. Guo, BTM: topic modeling over short texts, IEEE Trans. Knowl. Data Eng. 26 (12) (2014) 2928–2941.