ExtRA: A Framework for Extracting Prominent Review Aspects from Online Customer Feedback

***Abstract*—A popular way of analyzing and summarizing customer reviews about a product or service is to grade them based on a number of distinct review aspects. Traditionally, the *prominent review aspects* of a kind of product are determined manually but this costly approach does not scale apparently to various and emerging product types on large e-commerce platforms such as Amazon.com and Taobao.com or general review sites like Yelp.com. In this paper, we focus on mining the K most prominent aspects for a kind of product or service in the vast amount of review text data. We propose an automatic and unsupervised framework, ExtRA, for extracting the K most prominent aspect terms from massive amount of textual user reviews. This general framework can be applied to reviews of any types of product and service. Extensive experiments shows that our framework is effective and achieves the state-of-the-art performance for a diverse set of products and services.**

I. Introduction

Online user review is an essential part of e-commerce. Popular e-commerce websites feature enormous amount of text reviews, especially for popular products and services. To improve the user experience and expedite the shopping process, many websites provide either qualitative or quantitative summary of user reviews, typically organized by important aspects or characteristics. For example, Fig. 1 shows a short review passage from a customer on TripAdvisor.com, and the customer is also asked to give discrete ratings on several specific aspects of the hotel, such as location and cleanness. Such review summarization with specific aspects is defined as *aspect-based review summarization* [1], which more efficiently represents the customer feedback and thus is desired by both sellers and potential customers.

With aspect-based reviews summarization, potential customers can touch more details with various aspects of a product directly. They do not have to read a large amount of reviews and then make an evaluation considering different dimensions by themselves. Also, aspect-based review summarization provides an effective and efficient way for comparing similar products from different brands, since they share the same review aspects and then can be compared with each other in multiple dimensions. At present, aspect-based review summary offered by e-commerce typically falls into two main types: 1) manually created aspects and 2) automatically mined product-specific phrase sets. We discuss these two types in the following two paragraphs.

Some websites that only sell (or review) a very small number of products and services tend to manually select the most important aspects of their target products. For example, TripAdvisor.com only features travel-related products, and Cars.com only reviews automobiles. It is true that in-depth knowledge from human is the best way of characterizing a product type with a few keywords and the desired size of *prominent aspect terms* are indeed small enough for human to manually choose them. Nonetheless, as for many popular general e-commerce platforms, such as Amazon, ebay, Taobao and Yelp, there are a large amount of product types. Plus, new product types and service types are emerging almost everyday. Therefore, human labor would be too expensive and impractical to attack this task and it can also be difficult for human to predefine aspect terms for some emerging products. The key aspects of a product may also change in time. For example, in the past, people care more about the screen size and picture resolution of cell phones. These are no longer issues in present days. People instead turn to issues such as battery life and processing speed, etc.

Fig. 1: User review from TripAdvisor.

In general e-commerce websites, a common example of automatic review summarization is shown in Fig. 2. There are a set of automatically mined phrases for each phone model, along with the frequency of each phrase mentioned in the reviews. However, there are two drawbacks in this form of

summarization: 1) the set of phrases are from reviews of a specific product, and thus summaries appear to be different across different products within the same category, which makes comparison between products on a set of shared features difficult and unintuitive; 2) the emotional terms in the phrases would cause that potential customers cannot computationally see a broader level of reviews or computationally compare different products.

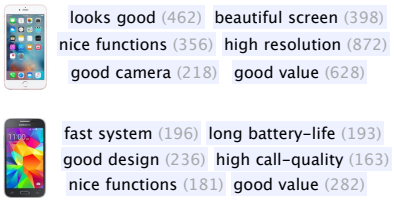


Fig. 2: Automatic review summarization for two mobile phones on an e-commerce website

Our goal in this paper is to develop an unsupervised framework for automatically extracting K prominent review aspects from user review corpora for any given product type. There are three main factors that make review aspect extraction more challenging:

- In aspect extraction, we expect the extracted K best aspects are important and representative enough for the given product or service, yet have little semantic overlaps with each other, so as to avoid very general and vague terms.
- The expression of opinions can be very versatile. In user reviews, aspects can appear both explicitly by direct references or through users' personal experiences implicitly.
- It is very common that opinions about multiple aspects are stated within a very short piece of comment, so the topics may shift from sentence to sentence much faster than other short documents.

Most previously proposed unsupervised approaches for aspect extraction are variations of topic modeling techniques. The main problem with topic-model-based methods is they typically leverage only word frequencies and co-occurrences information instead of semantics, and thus they cannot effectively detect the sentences that are superficially different but semantically talking about similar aspects.

In our framework, ExtRA, we perform topic modeling on sentence clusters to generate potential aspect topics, and cluster the potential topics to obtain the final pured aspect clusters. We propose a ranking mechanism including two ranking metrics respectively for pruning aspect clusters and ranking candidate aspect terms. Benefiting from the ranking mechanism, aspect clusters are properly represented as vectors (AspVec) in a shared space with aspect terms. Finally, our framework can then extract K most prominent aspects, including words and phrases, by simply computing the similarities between each AspVec and all the candidate aspect terms (words or phrases).

Our work has four main contributions:

1) We proposed a novel unsupervised framework for K[1] prominent aspects extraction from customer review corpus about any given product or service type.
2) Extensive experiments showed that our framework is effective and outperforms the state-of-the-art methods.
3) We open-sourced a tool of our framework and an evaluation dataset for future aspect extraction research.

In Sec. II we introduce our framework ExtRA step-by-step. Then, in Sec. III we evaluate our framework on multiple domains, demonstrate the effectiveness of our model against other approaches and show how the extracted aspects can be used to construct a complete review summarization. Finally, in Sec. IV, we discuss and compare our work with previous research on aspect-based review summarization.

II. ExtRA Framework

The review aspect extraction problem aims to extract K noun words or phrases from user reviews about a given type of product, each of which should represent a distinct aspect or feature of particular product or service type. Here K is an constant parameter for the problem. The set of reviews and the number of aspects are inputs. The overall workflow of ExtRA framework is shown in Fig. 3 which consists of 5 stages. We will discuss the motivation and details of each stage in following subsections.

A. Stage 1: Sentence Representations and Clustering

A typical hotel review extracted from Fig. 1 is shown below. We can see that topics (underlined) in this review can shift very quickly, and the adjacent sentences may refer to completely different aspects about a product. Plus, sentences about the same review aspect may not appear consecutively in a review.

> *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."*

Such fine-grained semantic shifts in user reviews make it difficult to simply apply the bag-of-word and normal topic modeling on reviews. Therefore we propose to work on sentence level instead of document level, and it would be helpful if we can divide long reviews with multiple aspects into several centered topic-oriented review segments.

Motivated by this observation, the first stage of our framework is to perform sentence clustering. Instead of using simplistic methods like bag-of-word representation, we leverage distributed representation which encodes words and sentences as low-dimensional real-valued vectors. Existing sentence representation methods include averaging word vector in a sentence, ParagraphVector (PV) [2], LSTM-based RNN [3] trained for language modeling etc.

After obtaining sentence representations, we cluster them into N clusters by k-means algorithm [4]. As a result, we obtain N clusters of sentences sharing similar semantics.

[1] K is usually less than 10 as the illustration example shown in Fig. 1

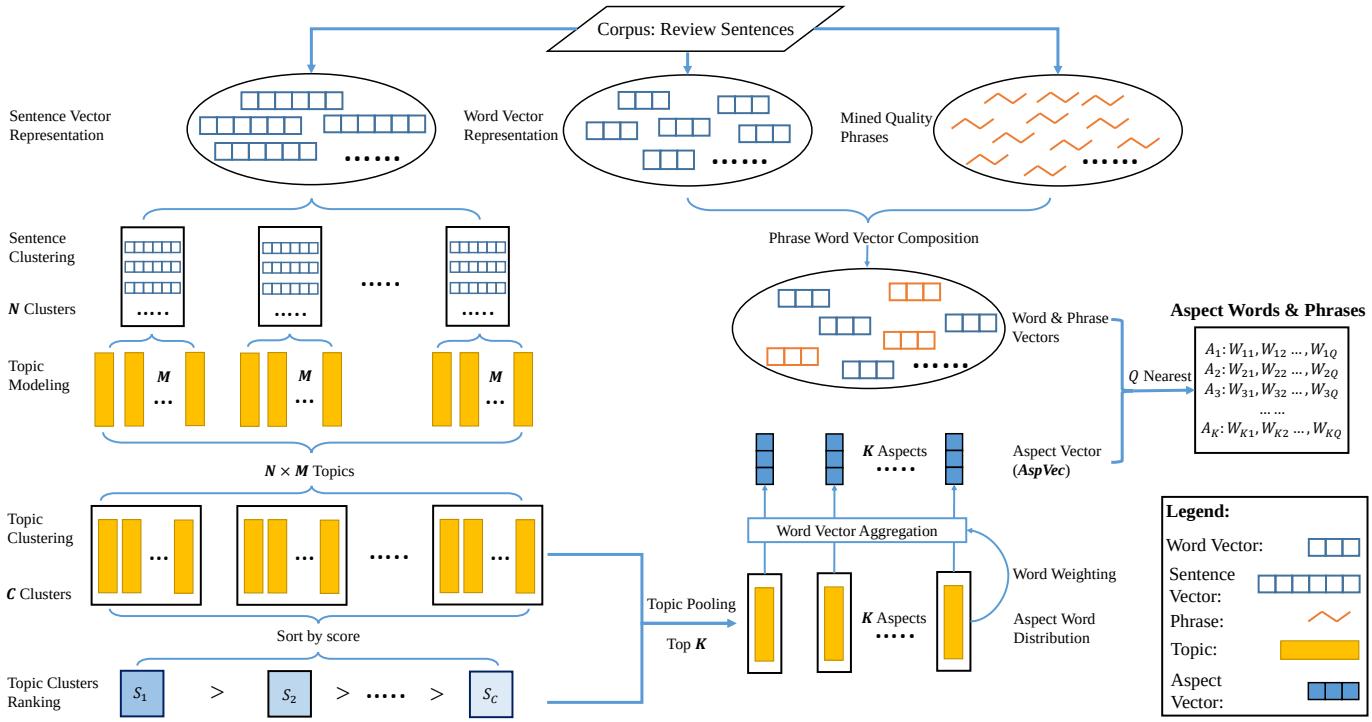


Fig. 3: Our overall framework.

B. Stage 2: Topic Modeling

Although we performed sentence clustering based on semantic sentence representations, there exit some sentences in the obtained cluster that may still mix up with some noise and overlapping aspects. For example, the following two sentences are taken from two TripAdvisor reviews. Sentence A) mentions multiple aspects such as *room*, *staff* and *price*. Sentence B) is only about the *location* aspect however lexically it seems to be talking about food. Thus, it is also difficult for sentence representations like PV to correctly determine the aspects in these sentences. As a result, each cluster contains multiple aspects and there are also overlapping across clusters.

> A) *"The room was clean, the staff were friendly, and I would say the price is very reasonable given the proximity to business and leisure destinations around downtown."*
>
> B) *"There is a restaurant just 5 min walk away with nice Italian food, pizza was great."*

To isolate the aspects and resolve such overlaps between short review pieces, we propose to perform traditional topic modeling in each cluster of sentences. We apply the state-of-the-art Biterm Topic Model (BTM) [5] topic modeling within each sentence cluster and generating M smaller topics for each cluster. This will give us in total $N \times M$ topics (word distributions). BTM is a word co-occurrence based topic model that learns topics by modeling word-word co-occurrences patterns (i.e., biterms). In contrast, LDA and PLSA are word-document co-occurrence topic models, since they model word-document co-occurrences. A biterm consists of two words co-occurring in the same context, for example, in the same short text window. Unlike LDA models the word occurrences, BTM models the biterm occurrences in a corpus. In generation procedure, a biterm is generated by drawn two words independently from a same topic. In other words, the distribution of a biterm $b = (w_i, w_j)$ is defined as:

$$P(b) = \sum_k P(w_i|z) * P(w_j|z) * P(z). \quad (1)$$

With Gibbs sampling algorithm, we can learn topics by estimate $P(w|k)$ and $P(z)$.

Table I shows an example of topics inferred from $N = 3$ sentence clusters with $M = 5$ from hotel reviews and illustrates the overlap problem. In this example, five topics (t1-t5) were extracted from each sentence cluster, and each row is one topic. It can be seen that the aspects for those three clusters should be *room*, *location* and *price* respectively. However, topics shown in boldface font obviously belong to other clusters. Especially, the last topic of the third cluster appears to be an overlap of more than two clusters. This topic modeling extracts multiple topics from each cluster, and each topic indicates the potential expected aspect.

C. Stage 3: Topic Clustering

To address the above problems, we propose to take a step forward and cluster the topics across the sentence clusters, which means we are supposed to cluster the previous $N * M$ clusters into C topic clusters. Here C is purposely set to be larger than K for that we would like to pruning them into K aspects later. We represent each topic as a topic vector which

TABLE I: Topics extracted from three sentence clusters of hotel review.

Sentence cluster 1	t1: room bed bedroom size floor t2: bedroom room wall size decor t3: room bathroom shower water towel t4: room suite size view floor t5: room shower area kitchen bed
Sentence cluster 2	t1: station minute tube location bus t2: location price night place rate t3: location square station street subway t4: distance bus subway downtown shopping t5: **restaurant city food buffet place**
Sentence cluster 3	t1: price rate service money star t2: **location city star time rate** t3: price service night money city t4: price location place night city t5: **location service food price restaurant**

is actually the weighted sum of word vectors using $p(w|t)$ as weights, among top k dominant words with highest $p(w|t)$ probabilities. Such topic vectors represents the topic centers by integrating the topic word semantics. Thus, the noisy topics can be clustered together and later discarded. We evaluate the quality influence of these redundant clusters on final aspects in our experiments.

TABLE II: Aspect clusters extracted from hotel reviews. Each row shows the candidate words of an aspect, sorted by the weight of each word.

breakfast, meal, food, tasty, dinner, morning, coffee, tea
room, night, time, bed, day, bathroom, staff, area, place
staff, desk, service, friendly, reception, concierge, helpful
close, city, location, place, central, station, bus, street
bed, shower, spacious, room, size, bathroom, bedroom, floor
price, room, check, night, money, city, location, star, service
location, price, room, night, place, rate, money, time, city

Finally, for each cluster (e.g. cluster c) we take the mean of the $(N \times M)/C$ topics (shown in Eq. (2)) and normalize it (shown in Eq. (3)) for the word distribution of that cluster. We call them *aspect clusters*. In Eq. (2) and Eq. (3), w represents the word in cluster c, W_c represents the set of words in cluster c, T_c is the topic set of c and $p_c(w)$ denotes the significance score of word w in cluster c, which is the sum of the weights of w from topic t (e.g. $v_t(w)$) in cluster c.

$$m_c(w) = \frac{\sum_{t \in T_c} v_t(w)}{|T_c|} \tag{2}$$

$$p_c(w) = \frac{m_c(w)}{\sum_{w' \in W_c} m_c(w')} \tag{3}$$

Some example aspect clusters extracted from hotel reviews are shown in Table II.

D. Stage 4: Aspect Cluster Ranking

In previous steps, we formed C aspect clusters including some noisy clusters supposed to be removed. We propose a *distinctiveness score* to measure the cluster quality which captures the intuition that the more the cluster overlaps with others the lower distinctiveness score associates to it. c ($c \in [1, C]$) indicates the cth cluster, the distinctiveness score S_c is defined as following:

$$\begin{aligned} S_c &= \sum_{w \in W_c} S_c(w) \\ &= \sum_{w \in W_c} \log \left(\frac{p_c(w)}{1 + \sum_{c' \neq c} p_{c'}(w)} \right), \end{aligned} \tag{4}$$

where $p_c(w)$ is the significance score in Eq. (3). Subsequently, the clusters are ranked in the descending order of this score and the last $C - K$ clusters are discarded. The result is shown in Table III, the discarded cluster are grayed-out.

TABLE III: Aspect clusters ranked by distinctiveness score. Potential aspect words are boldfaced.

staff, desk, **service**, friendly, reception, concierge, helpful
breakfast, meal, **food**, tasty, dinner, morning, coffee, tea
price, room, check, night, money, city, location, star, service
bed, shower, spacious, **room**, size, bathroom, bedroom, floor
close, city, **location**, place, central, station, bus, street
room, night, time, bed, day, bathroom, staff, area, place
location, price, room, night, place, rate, money, time, city

E. Stage 5: Aspect Generation

After previous aspect cluster ranking stage, we obtain K aspect clusters, each of which represents a potential aspect. We aims to extract top aspect candidates from those clusters. We propose two methods toward this goal, 1) a work ranking algorithm which only produces a list of aspect words, 2) an embedding based aspect term encoder-decoder algorithm that uses a vector representation known as AspVec and produces both words and phrases.

1) Word Ranking: This algorithm indicates the most prominent words in each aspect cluster. After such ranking, we can simply select the top-ranked word as aspect words. We refer this method as *ExtRA-Words*.

In Table III, we manually boldfaced the most representative words for each cluster. Each of them is the expected aspect word which can be treated as a summary of other words in the same cluster. However, it can be seen that not all of them have the highest probability in their clusters. In order to automatically select the best aspect words, we propose a word ranking step to adjust the order of word in aspect cluster by considering both the weights and semantics of words. We design our ranking metric to capture prominence of words in aspect cluster. Intuitively, the most prominent and representative word in each cluster is assumed to be the closest to centroid of the aspect cluster. Therefore, we calculate the semantic similarity of each word with all other words in the cluster and use that to measure how central that word is. To calculate the semantic similarity between two words, we use the cosine similarity of word embeddings. Word2Vec is a very popular word embedding model and has been shown to perform well on the semantic similarity tasks [6].

In order to prevent generating duplicate aspects from different aspect clusters, we process the clusters in a sequential order, based on the ranking given by the previous step of aspect cluster ranking. When we calculate the score for word x in cluster i, we also consider the scores of word x in cluster 1 to $i-1$, where the scores have already been calculated. We prevent the duplicate aspect words by subtracting the scores of other clusters from the score of cluster i. The score of word x in the ith cluster i is thus defined by:

$$s_c(x) = u_c(x) \sum_{y \in W_c} \hat{x} \cdot \hat{y} - \sum_{c'=1}^{c-1} s_{c'}(x), \tag{5}$$

where $\hat{x}$ is the vector representation of x; $u_c(x)$ is the weight of x in cluster c. The words in each cluster is ranked by this score following the order given by cluster ranking. This gives us the final aspect clusters. The effect of duplicate prevention is evaluated in our experiments.

After word ranking step, each word x associates a representative score $s_c(x)$ in each aspect cluster c. Such word representative score considers both the distinctiveness of cluster c and the centeredness of x inside c.

2) Aspect Term Encoder-Decoder: In order to consider multi-word aspects, we extend our vocabulary with high quailty phrases extracted by AutoPhrase [7]. We propose a method to encode words, phrases and aspect clusters into the same vector space. Therefore, we can compute the similarity between them with ease.

We encode aspect clusters as vectors named *AspVec*, by taking top T ($T = 50$) most representative words in each aspect cluster and summing up their embeddings weighted by corresponding normalized word ranking score $w_c(x)$. *AspVec* integrates the prominent word semantics together, representing the semantic of aspect cluster as following:

$$w_c(x) = \frac{s_c(x)}{\sum_{y \in V_T} s_c(y)} \tag{6}$$

$$AspVec(c) = \sum_{t=1}^{T} w_c(x) E(x), \tag{7}$$

where c is the aspect cluster, V_T is the vocabulary of top T words, x, y are words in V_T, and $E(x)$ is the embedding of x.

We also encode the phrase (p) vector representation ($E(p)$) in the same vector space, averaging the embedding of words in the phrase.

$$E(p) = \sum_{x \in p} E(x) \tag{8}$$

In such way, we encode the aspect cluster centers and surface terms including words and phrases into the same vector space. It is natural to select terms nearest to cluster center as the aspects. Now aspect candidates include both words and phrases. We find the semantically nearest words or phrases for the aspect cluster in order to decode $AspVec$ into representative surface terms. Therefore, we search Q-Nearest neighborhoods for particular aspect cluster, i.e. $AspVec$, harvesting a ranking list. We pick the top-ranked candidate as the final extracted aspect.

III. Experiments

In this section, we present the results of three experiments. The first one is a comparison of methods for obtaining sentence vectors. In the first step of our framework, i.e., sentence clustering, sentence representation is crucial to the performance and hence critical to overall framework. In particular, we need a vector representation that accurately captures the semantic similarity between sentences, so we test the performance of the methods on a sentence similarity prediction task. The second experiment is the comparison of our framework with a number of baselines including the state-of-the-art approaches in the end-to-end aspect extraction task. The third experiment is a complete aspect-based summarization application using our model.

A. Experiment 1: Comparison of Sentence Vectors

The vector representation of sentences is a vital module of our framework, this experiment compares two models: long short-term memory (LSTM) and paragraph vector (PV).

Since we use the sentence vectors for clustering the sentences based on their topics, and k-means algorithm uses the distances between the vectors to determine which cluster each sentence belongs to, we need the sentence vectors to accurately capture the semantic distance between the sentences. For this purpose, we evaluate the models using a sentence similarity prediction task, namely the English subtask of SemEval-2015 Task 2: Semantic Textual Similarity [8]. There are 14,250 data entries, 11,250 for training and 3000 for test, each containing a pair of sentences and a score from 0-5 rating the semantic similarity of the two sentences. We train our models on all the sentences and test on the 3000 sentence pairs from the test set.

For each sentence pair, both neural network models give us the independent 300-dimensional sentence vectors; to use this pair of vectors to predict the semantic similarity, we train an independent predictor backend. The backend is a three-layer feed-forward neural network. It takes the two vectors for the sentence pair given by LSTM or PV as input. The input layer takes the concatenation of the two 300-dimensional vectors; the hidden layer is 100-dimensional; the output layer is a 6-class softmax, corresponding to the scores of 0 to 5; the loss function is categorical cross-entropy. The backend is trained independently for both LSTM and PV, with the same input dimensionality and network structure, and for the same number of iterations. To put our results in context, we also include the results of the winning system of SemEval-2015 task 2, DLS@CU [9]. Note that DLS@CU doesn't use sentence vectors and cannot be used in our framework. The results are shown in Table IV.

We can see that, although with fewer parameters, PV outperforms LSTM on this task. This is probably because the last hidden vector of LSTM is not a proper representation of the sentence for the semantic similarity task, due to its emphasis on the last seen word as opposed to earlier words.

TABLE IV: Pearson correlation scores on 5 parts of the test set and the mean scores.

Model	Test 1	Test 2	Test 3	Test 4	Test 5	Mean
LSTM	0.5717	0.6329	0.6015	0.7961	0.8147	0.6833
PV	0.5928	0.6681	0.6292	0.8044	0.8419	0.7072
DLS@CU	0.7390	0.7725	0.7491	0.8250	0.8644	0.8015

The performance of different sentence vectors is included in the next experiment.

B. Experiment 2: End-to-end Evaluation of K Best Aspects Extraction

1) Data: The data we use in this experiment was gathered from various e-commerce websites, including amazon.com, tripadvisor.com, etc. The reviews are about 6 categories of product or service. The review content is plain English text and we do not use any label for training our model. We use human labels for evaluation. The product categories, their sources and the sizes of the review datasets are summarized in Table V.

TABLE V: Dataset summary.

Product type	Source	No. of Reviews
hotel	TripAdvisor	27145
mobile phone	Amazon	3716
mp3 player	Amazon	2745
laptop	Amazon	5471
cameras	Amazon	3077
restaurant	Yelp	4016

To benefit the downstream applications such as aspect-based review summarization shown in Sec. III-C, we expect the K extracted aspects are most important and representative for the given kind of products or services. The existing published dataset [1], [10]–[12] collect fine-grained features and aspect terms from review sentences which are not prominant enough to evaluate our system. Therefore, for each category, we ask 5 annotators who are proficient with English to annotate 5 different words that cover the most prominent aspects of the corresponding product or service category. Our annotated ground truth are all words by design, since the multi-word expressions for aspects are very versatile. Therefore, we do not show the accuracy of our ExtRA-PV+Phrase model (mentioned in III-B3d) in Table VIII. The labels provided by the 5 annotators are aggregated together without removing duplicated words, so we have 25 words in total.

When evaluating the models, we compare the 5 aspect words generated by the models with those provided by the annotators. We calculate the portion of words among the 25 labels that are correctly generated by the model as the accuracy of the model. The ground-truth is shown in Table VI.

2) Parameter Tuning: In this section we conduct experiments to determine the best set of parameters of our model, most importantly the constants N, M, C.

Number of Sentence Clusters (N)

In the first clustering step we use a k-means to cluster the sentences, where each cluster should contain sentences about similar aspects. We need to determine the number of clusters, N. We use an empirical elbow method to determine N, we plot the total distance between each point and its corresponding center, a.k.a. the loss, with different choices of N, as shown in Fig. 5. Note that we don't concern about the number of expected aspects in this experiments. As we can see from Fig. 5, to give the lowest loss, the number of sentence clusters should be 10 or 11. Therefore, in the following experiments, we set N to be 10.

TABLE VI: Ground-truth. Each row is provided by one annotator.

hotel	room price location service utility room service price food location sleep service room price location location price bedroom bath staff room price bath staff location
camera	image lens battery memory carry picture lens price battery mode image price battery design operation image lens battery focus storage image appearance lens portability battery
restaurant	location price food service cleanness food price location environment service price food quietness location staff food price service environment location food price location service environment
mobile phone	brand price quality battery screen price quality camera touch battery quality price design screen carry quality price OS battery service price quality screen battery color
mp3 player	price quality sound screen battery carry price design sound screen price quality carry earphone sound quality price battery sound carry price quality sound carry screen
laptop	price quality brand OS battery quality price battery memory CPU disk memory CPU screen keyboard price battery screen CPU performance quality price appearance battery keyword

Number of LDA Topics (M)

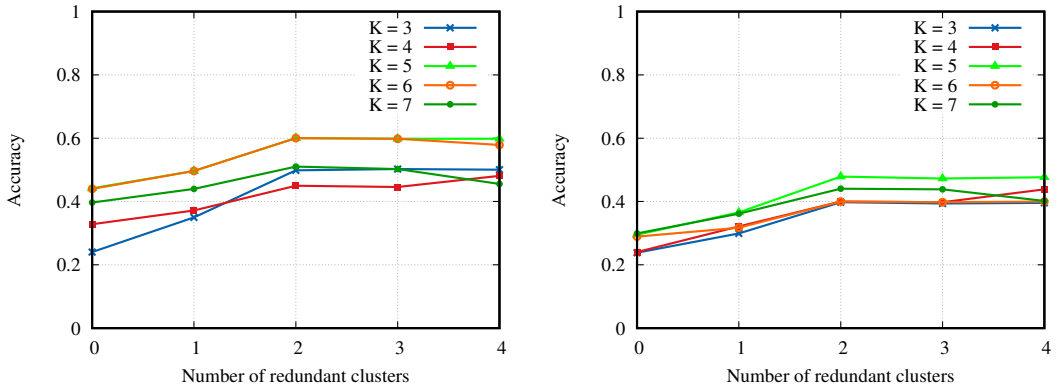
LDA in our model serves the purpose of isolating the noisy topics and resolving the overlaps between aspects. In our experiments we find that different number of LDA topics doesn't have much effect on the end-to-end performance. As an example, the aspect extraction accuracy with different M on hotel reviews are shown in Table VII. Consider the 25 manually annotated words again, the difference between the three numbers is only 1 out of 25 words. In our experiments, M is fixed to 10.

TABLE VII: The effect of different number of LDA topics

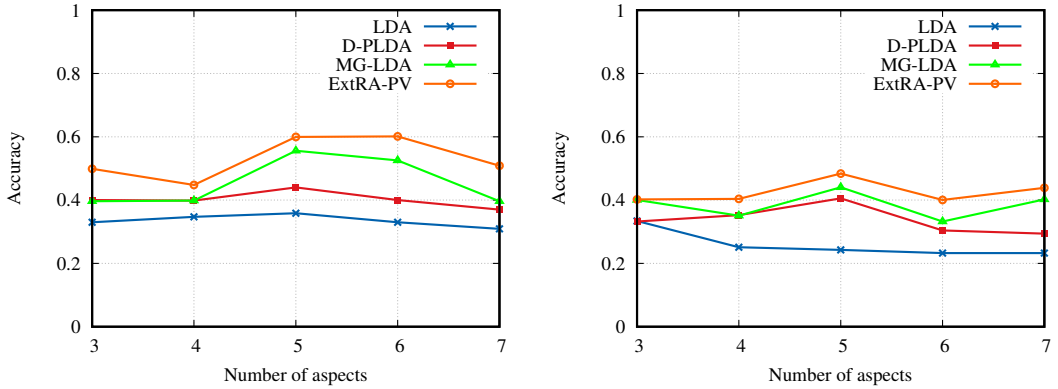
M	5	8	10
Accuracy	17/25	16/25	17/25

Redundant Clusters (C)

As mentioned in Sec. II-C, we generate C aspect clusters before ranking the clusters and the words, where C is larger than K. In this section we demonstrate the effect of these redundant clusters. In Fig. 4a, we show the accuracy with



(a) The performance of different models when adjusting the number of redundant aspects $C - K$. Left: hotel reviews. Right: mobile phone reviews.



(b) The performance of different models when adjusting the number of expected aspects K. Left: hotel reviews. Right: mobile phone reviews.

Fig. 4: Parameter Selection

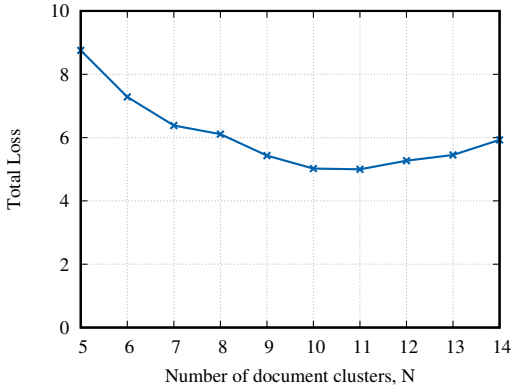


Fig. 5: Selecting N, the number of clusters for k-means. The optimal N is selected to be 10.

different redundancy $C-K$, ranging from 0 to 4, with number of expected aspects ranging from 3 to 7. It can be seen that 2 redundant clusters are sufficient and more redundancies can't improve the performance.

3) Models: In this section we introduce the models used in our end-to-end comparison. We compare 5 models in our experiments, LDA as a simple baseline, D-PLDA [13] as a representative for joint aspect-sentiment models, MG-LDA [14] as a representative for aspect extraction topic models, and two variations of our model, i.e. ExtRA-LSTM and ExtRA-PV. We run the 5 models on the review data for each product type separately. The number of aspects is fixed to 5.

a) LDA: We use LDA as a simple topic model baseline. We treat each review as a document and run on the whole corpus (single product type). The number of topics is set to 5 for model comparison.

b) D-PLDA: D-PLDA [13] is an LDA variation designed specifically for modeling user reviews. In D-PLDA, only opinion phrases are modeled, and the nouns and the phrases are controlled by two separate hidden parameters. There is a dependency from hidden parameter for adjectives to the one for nouns. We use D-PLDA as a representative for models with join aspect and sentiment inference.

c) MG-LDA: To compare our model with a popular, well-performed model designed for aspect extraction, we use MG-LDA [14]. MG-LDA also models topics at different granularities. For model comparison, the number of local topics (aspect topics) is set to 5, the number of global topics is set to 30 (ignored in evaluation).

d) Our models: For sentence vectors (both LSTM and PV), the dimensionality is set to 300. We refer such models as ExtRA-PV and ExtRA-LSTM. Our framework can be extended to extract multi-word aspects by integrating AspVec, which is called ExtRA-PV+Phrase. The number of sentence-level clusters is set to 10; the number of LDA topics is 10; the number of final word clusters in the clustering phase is 7 and is later reduced to 5 in the ranking phase. For training the sentence vectors, we do not use any extra data, all the word and sentence vectors are trained on the set of reviews for a single product type.

TABLE VIII: Comparison of accuracies using different models for aspect extraction.

Type \ Model	LDA	D-PLDA	MG-LDA	ExtRA-LSTM	ExtRA-PV
hotel	0.36	0.44	0.56	0.56	**0.68**
mobile phone	0.40	0.48	0.60	0.60	**0.72**
mp3 player	0.28	0.40	0.48	0.44	**0.52**
laptop	0.32	0.44	0.60	0.56	**0.64**
camera	0.34	0.43	0.62	0.55	**0.62**
restaurant	0.24	0.40	0.52	0.48	**0.60**

4) End-to-End Comparison: In the end-to-end evaluation, we compare the performance of our models on aspect extraction with three other models as mentioned above. The results are shown in Table VIII. Our models outperform others in all of the categories. Our model with paragraph vector performs better than with LSTM, which is consistent to our result from the previous experiment.

We show the aspect word clusters on hotel reviews in Table IX. The top words, which are chosen as the representatives of the aspects, are shown in boldface. Fig. 6 is the visualization (using TopicCloud toolkit[2] [15]) for aspect clusters generated by ExtRA-PV+Phrase model on hotel review. The portion of area for clusters depends on their distinctiveness score. And the score of words in each cluster determines their size. With this visualization, the importance of each cluster and word is clearly shown, which makes comprehension easier.

5) Effect of Number of Expected Aspects (K): To evaluate the effect of different number of expected aspects, i.e. K, which determines the coverage covered by expected aspects, we ask our annotators to provide different number of labels on two categories. The number of aspects range from three to seven. The performance of the four models when changing the number of aspects are shown in Fig. 4b. It can be seen that our models can well extract aspects with variant coverage and perform constantly better than all other models regardless of the number of expected aspects.

6) Effectiveness of Ranking: Here we test the effectiveness of several techniques proposed for the ranking step. In particular, we compare three setups:

- No word ranking. Use directly the word with highest score from aspect cluster after aspect inference. Remove redundant clusters with cluster ranking.
- No duplicate prevention. As mentioned in Sec. II-E, we do duplicate prevention by considering other clusters when ranking words within a cluster. In this setup we do not include this effort.
- Use all ranking methods.

The results are shown in Table X. The ranking scheme we propose, that is, word ranking with duplicate prevention, has clear advantage across all product categories.

[2]Open sourced toolkit is available at https://github.com/askerlee/topiccloud

Fig. 6: Aspect Cloud: visualization of aspect words and phrases extracted from the hotel reviews.

TABLE IX: Top aspect words (with phrases) for hotel reviews by different models

ExtRA-LSTM	**service**, front, desk, reception, concierge, check, gust **location**, station, minute, tube, station, bus, distance **time**, check, day, desk, charge, book, front, hour, night **food**, coffee, buffet, morning, tea, room, fruit, egg, juice **bed**, bathroom, size, floor, view, suite, king, book, decor
ExtRA-PV	**staff**, service, room, front, desk, check, concierge **food**, breakfast, bar, restaurant, coffee, morning, tea **price**, parking, night, place, rate, service, money, star **room**, bed, bathroom, size, suite, floor, view, bedroom **location**, minute, square, subway, street, block, distance
ExtRA-PV+Phrase	**staff**, front_desk, check, custom_service, reception, concierge **breakfast**, coffee, buffet, food, fruit, juice, tea **location**, minute, walking_distance, bus, street, block **room**, bed, bathroom, shower, air_conditioning, view, size **swimming_pool**, pool, guest, place, service, day
MG-LDA	**shower**, bathroom, room, floor, area, bedroom, desk, tea **time**, day, room, check, front, desk, night, service **food**, bar, service, breakfast, restaurant, staff, taxi **room**, bed, floor, place, air, night, bathroom, noise **price**, business, service, star, internet, location, staff
DP-LDA	**service**, front, desk, reception, concierge, check, guest **station**, minute, tube, location, bus, distance, street **check**, day, time, desk, charge, book, front, hour **coffee**, buffet, morning, tea, room, day, fruit, food **bed**, bathroom, size, floor, view, suite, king, book
LDA	**stay**, night, place, trip, time, weekend, night, hour **location**, square, street, place, restaurant, market, block **room**, bed, bathroom, size, tv, king, suite, pillow **staff**, service, desk, location, concierge, room, night **room**, floor, noise, view, night, water, door, bathroom

TABLE X: The accuracy performance with different ranking setups.

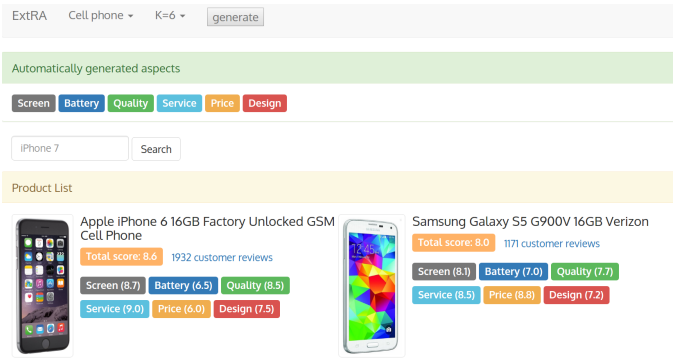
Type \ Setup	No word ranking	Word ranking w/o. duplicate prevention	Word ranking w. duplicate prevention
hotel	0.48	0.60	**0.68**
mobile phone	0.52	0.64	**0.72**
mp3 player	0.36	0.44	**0.52**
laptop	0.44	0.52	**0.64**
camera	0.43	0.49	**0.58**
restaurant	0.48	0.56	**0.60**

C. Experiment 3: Aspect-based Review Summarization

In the last experiment, we present an end-to-end demo system based on our proposed aspect extraction framework ExtRA. User can select a product type from a predefined set of categories, and sets the number of aspects K. Fig. 7a shows an example about mobile phones. The aspect clusters by our method on the mobile phones review data is shown in Table XI. We use the top aspect words extracted by our model to represent clusters (in this example, we set $K = 6$). The user may filter out some of the results by using the search box. The system demonstrates the review summaries of the selected products. Each summary contains the aspects and their scores. The scores are calculated according to the sentiment value of review sentences. To identify the aspects in the review sentences, we check if the top aspect words in each cluster appeared. If so, the sentiment value of the sentence for the identified aspects is predicted with an LSTM and feed-forward network model trained on Stanford Sentiment Treebank [16]. The score for each aspect from the whole set is the average of the predicted sentiment values attached on it. With this system, users can easily compare different products with respect to the same set of important aspects, without having to read a lot of reviews. Comparison between Apple iPhone 6 and Samsung Galaxy S5 (see Fig. 7a) clearly indicates that Apple has better quality at the cost of less competitive price. Besides this concise format, sometimes users need much more details before their final decision. For this purpose, if the user wishes to understand why a product receives a score for a certain aspect, they can click on the "xxx customer reviews" link and be directed to the original review snippets with the aspect words and sentiment words highlighted (annotated using Stanford CoreNLP toolkit [17]). The review snippets can be further grouped by aspects by clicking on the tabs(see Fig. 7b).

TABLE XI: The aspect clusters generated from mobile phone reviews

screen, resolution, touch, display, color, picture
battery, power, charge, day, cable, charger
quality, break, day, build, buy, control
service, buy, check, help, website, shipping
price, money, worth, cost, charge, free
design, color, metal, case, plastic, silver



(a) Automatically generated 6 aspects for mobile phones

(b) Review snippets displayed by the aspects

Fig. 7: Demo system for ExtRA

IV. Related Work

There is much existing work on aspect extraction and they can be roughly divided in two categories. The first kind is predominately rule-based methods that find the aspect candidates first and then cluster them. The second kind are mostly topic model approaches, which cluster the aspect candidates and other words together, then identify the aspects.

Rule-based methods rely heavily on parsing and the quality of parsing. Based on the parsing result, they use features such as frequency and modifying relationship or a set of manually defined rules to identify the aspect candidates. A problem with this kind of method is *implicit aspect extraction*, that is to find those aspect candidates that are expressed without directly mentioning the aspect word. For example in a mobile phone review "Under heavy usage it will last at least a day" is about the aspect of battery life. Also co-reference may be a problem as in this hotel review: "I really liked the room. It is large and comfy." Rule-based methods need carefully designed rules or other methods for implicit aspect extraction. Poria et al. 2014 [18] uses manually crafted mining rules. Qiu et al. 2011 [19] also used rules, plus the Double Propagation method to better relate sentiment to aspects. Gindl et al. 2013 [20] also used the Double Propagation method, with the help of anaphora resolution for identifying co-references to improve

the accuracy. Su et al. 2008 [21] used a clustering method to map the implicit aspect candidates (which were assumed to be noun form of adjectives in the paper) to explicit aspects. [22] mapped implicit features to explicit features using a set of sentiment words and by clustering explicit feature - sentiment pairs.

Topic models have been used to perform extraction and clustering at the same time. Most existing work are based on two basic models, pLSA [23] and LDA [24]. Applying topic models on user reviews requires extra attention to the nature of review texts. Two features of review texts are often considered in several modifications of topic model. The first feature is the quick shift of topics between sentences, since people express multiple opinions about various aspects within a short piece of text. Sentences close to each other may talk about completely different but related topics. Phrase-based LDA The other feature is the prominence of sentiment. Since reviews express opinions, naturally there are many sentiments, and also there is a strong relationship between the sentiments and the aspects. Many variations of LDA exploits this feature to improve the mining of aspects. Lakkaraju et al. 2011 [25] models in parallel aspects and sentiments per review. Lin et al. 2009 [26] models the dependency between the latent aspects and ratings. Wang et al. 2011 [27] proposed a generative model which incorporates topic modeling technique into the latent rating regression model [28]. Moghaddam et al. 2012 [13] made a nice summarization of some basic variations of LDA for opinion mining. Our method can be thought of as a hybrid approach with topic modeling as one of its elements.

Most previous work on aspect extraction use variations of topic modeling, and aspects are modeled as topics, that is, word distributions. To our knowledge, all previous work requires manual selection to choose the best word as a representative for each topic. For example in Titov et al. 2008 [14], the authors manually labeled each topic inferred from mp3 player reviews. On the contrary, our method is designed to automatically select the best words so that the whole process requires no human effort or labels. We argue that this is an important difference for real-world application, since selecting the best words for each product category is still too much effort for websites like Amazon and Yelp, while our method requires no manual processing and thus can be used directly for real-world applications like aspect-based review summarization. Moreover, such automatic aspect extraction enables dynamic change of the aspect words over time to reflect changing customer interest or taste.

The prominent aspect keywords automatically extracted by our approach can be fed to downstream aspect-based review summarization frameworks. The aspect clusters generated by our model can be used for two purposes:

- The top word of each cluster can be used as the basis of review summarization.
- The clusters can be used for identifying aspects in review texts.

In Sec. III-C we used a simple neural network model to predict the sentiment score for each aspect. More sophisticated neural network models for single sentence sentiment prediction [16], [29]–[31] can be used. Together they can form an automatic chain of software to produce accurate, quantitative review summaries from massive online reviews.

V. Conclusion

In this paper, we propose our multistage framework ExtRA for extracting most prominent aspect terms from user reviews, which is beneficial for both qualitative and quantitative review summarization and opinion mining for any types of product or service. We found that simply performing topic modeling alone does not do very well with this problem because reviews are highly condense and tend to switch review aspects quickly within a short text. Our unsupervised framework solves the problem by slicing and shuffling the reviews and then clustering them in sentence level and topic level respectively. Finally we design our AspVec to represent the semantics of aspects and use it to extract aspect terms by computing the similarities. Extensive experimental results show that this approach outperforms other the state-of-the-art models.

As for the future work, we will improve AspVec to better represent the semantics of topic clusters. Also, we would generalize our framework to other domains, e.g., extracting answer aspects from community question answering.

References

[1] M. Hu and B. Liu, "Mining and summarizing customer reviews," in *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2004, pp. 168–177.

[2] Q. V. Le and T. Mikolov, "Distributed representations of sentences and documents," in *ICML*, 2014.

[3] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[4] S. P. Lloyd, "Least squares quantization in PCM," *IEEE Trans. Information Theory*, vol. 28, no. 2, pp. 129–136, 1982. [Online]. Available: https://doi.org/10.1109/TIT.1982.1056489

[5] X. Cheng, X. Yan, Y. Lan, and J. Guo, "Btm: Topic modeling over short texts," *IEEE Transactions on Knowledge and Data Engineering*, vol. 26, no. 12, pp. 2928–2941, 2014.

[6] O. Levy, Y. Goldberg, and I. Dagan, "Improving distributional similarity with lessons learned from word embeddings," *Transactions of the Association for Computational Linguistics*, vol. 3, pp. 211–225, 2015.

[7] J. Liu, J. Shang, and J. Han, "Phrase mining from massive text and its applications," *Synthesis Lectures on Data Mining and Knowledge Discovery*, vol. 9, no. 1, pp. 1–89, 2017.

[8] E. Agirrea, C. Baneab, C. Cardiec, D. Cerd, M. Diabe, A. Gonzalez-Agirrea, W. Guof, I. Lopez-Gazpioa, M. Maritxalara, R. Mihalceab *et al.*, "Semeval-2015 task 2: Semantic textual similarity, english, spanish and pilot on interpretability," in *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, 2015, pp. 252–263.

[9] M. A. Sultan, S. Bethard, and T. Sumner, "Dls@cu: Sentence similarity from word alignment and semantic vector composition," in *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*. Denver, Colorado: Association for Computational Linguistics, June 2015, pp. 148–153. [Online]. Available: http://www.aclweb.org/anthology/S15-2027

[10] A.-M. Popescu and O. Etzioni, "Extracting product features and opinions from reviews," in *Natural language processing and text mining*. Springer, 2007, pp. 9–28.

[11] J. Pavlopoulos and I. Androutsopoulos, "Aspect term extraction for sentiment analysis: New datasets, new evaluation measures and an improved unsupervised method," *Proceedings of LASMEACL*, pp. 44–52, 2014.

[12] X. Ding, B. Liu, and P. S. Yu, "A holistic lexicon-based approach to opinion mining," in *Proceedings of the 2008 international conference on web search and data mining*. ACM, 2008, pp. 231–240.

[13] S. Moghaddam and M. Ester, "On the design of lda models for aspect-based opinion mining," in *CIKM*, 2012, pp. 803–812.

[14] I. Titov and R. McDonald, "Modeling online reviews with multi-grain topic models," in *WWW*, 2008, pp. 111–120.

[15] S. Li and T.-S. Chua, "Document visualization using topic clouds," *arXiv preprint arXiv:1702.01520*, 2017.

[16] R. Socher, A. Perelygin, J. Y. Wu, J. Chuang, C. D. Manning, A. Y. Ng, and C. Potts, "Recursive deep models for semantic compositionality over a sentiment treebank," in *EMNLP*, 2013, pp. 1631–1642.

[17] C. D. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. J. Bethard, and D. McClosky, "The Stanford CoreNLP natural language processing toolkit," in *Association for Computational Linguistics (ACL) System Demonstrations*, 2014, pp. 55–60. [Online]. Available: http://www.aclweb.org/anthology/P/P14/P14-5010

[18] S. Poria, E. Cambria, L.-W. Ku, C. Gui, and A. Gelbukh, "A rule-based approach to aspect extraction from product reviews," in *Proceedings of the Second Workshop on Natural Language Processing for Social Media (SocialNLP)*, 2014, pp. 28–37.

[19] G. Qiu, B. Liu, J. Bu, and C. Chen, "Opinion word expansion and target extraction through double propagation," *Computational linguistics*, vol. 37, no. 1, pp. 9–27, 2011.

[20] S. Gindl, A. Weichselbraun, and A. Scharl, "Rule-based opinion target and aspect extraction to acquire affective knowledge," in *Proceedings of the 22nd international conference on World Wide Web companion*. International World Wide Web Conferences Steering Committee, 2013, pp. 557–564.

[21] Q. Su, X. Xu, H. Guo, Z. Guo, X. Wu, X. Zhang, B. Swen, and Z. Su, "Hidden sentiment association in chinese web opinion mining," in *Proceedings of the 17th international conference on World Wide Web*. ACM, 2008, pp. 959–968.

[22] L. Zeng and F. Li, "A classification-based approach for implicit feature identification," in *Chinese Computational Linguistics and Natural Language Processing Based on Naturally Annotated Big Data*. Springer, 2013, pp. 190–202.

[23] T. Hofmann, "Probabilistic latent semantic indexing," in *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 1999, pp. 50–57.

[24] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *The Journal of Machine Learning Research*, vol. 3, pp. 993–1022, 2003.

[25] H. Lakkaraju, C. Bhattacharyya, I. Bhattacharya, and S. Merugu, "Exploiting coherence for the simultaneous discovery of latent facets and associated sentiments," in *SDM*. SIAM, 2011, pp. 498–509.

[26] C. Lin and Y. He, "Joint sentiment/topic model for sentiment analysis," in *Proceedings of the 18th ACM Conference on Information and Knowledge Management*. ACM, 2009, pp. 375–384.

[27] H. Wang, Y. Lu, and C. Zhai, "Latent aspect rating analysis without aspect keyword supervision," in *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2011, pp. 618–626.

[28] ——, "Latent aspect rating analysis on review text data: a rating regression approach," in *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2010, pp. 783–792.

[29] A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts, "Learning word vectors for sentiment analysis," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*. Association for Computational Linguistics, 2011, pp. 142–150.

[30] C. N. dos Santos and M. Gatti, "Deep convolutional neural networks for sentiment analysis of short texts." in *COLING*, 2014, pp. 69–78.

[31] D. Tang, B. Qin, and T. Liu, "Document modeling with gated recurrent neural network for sentiment classification," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 2015, pp. 1422–1432.