Unsupervised Automatic Aspect Extraction from Online Reviews

Anonymous

***Abstract*—One popular way of summarizing users opinions about a product or service is to grade it by a number of distinct aspects shared by the same type of product or service. Traditionally, the aspects of a product type are determined manually but this doesn't scale to diverse product types on large e-commerce platforms or universal review sites such as Amazon.com, Taobao.com or Yelp.com. In this paper, we propose a unsupervised multistage approach for automatically discovering the best aspect words from massive amount of textual user reviews. This method can be applied to reviews of any product or service types. Our experiments showed that our approach is efficient and achieves the state-of-the-art accuracies for a diverse set of products and services.**

I. Introduction

Online user review is an integral 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 provides either qualitative or quantitative summary of the user reviews, organized by important aspects or characteristics of the target product or service. Two examples of such *aspect-based review summarization* [1] are shown in Fig. 1 and Fig. 2. In Fig. 1 from TripAdvisor, besides the short review passage written by the user, the user are asked to give discrete ratings (on the scale of 1-5) on various aspects of the hotel room, e.g., location and cleanness. The ratings of a product from individual reviews can then be aggregated into an overall ratings of the same product by many users, such as those shown about a specific car model in Fig. 2, a snapshot from Cars.com.

Fig. 1. User review from TripAdvisor.

Fig. 2. Review summarization from Cars.com.

Aspect-based reviews have several advantages compared to the more traditional style of online reviews that consists of a short passage and an overall rating. In aspect-based reviews, more details are provided quantitatively and more directly, and the users can learn about various aspects of a product without having to read the entire review passage. Another advantage of aspect-based reviews is that different products within the same category can be compared directly with respect to multiple aspects, instead of just an overall rating. When researching on products, users spend most of their time comparing different brands and models. Aspect-based review summarization provides an effective and efficient way for doing such comparison, saving the users both time and effort.

At present, websites that offer aspect-based review summaries typically only feature a single or a small number of product categories, e.g., TripAdvisor.com only features travel related products while Cars.com reviews automobiles. The reason is that it takes in-depth knowledge to be able to best characterize a product type using a few keywords, that are both relevant to the current user interests and diverse enough to cover as many facets of the type as possible. It is such a difficult task that these aspects are mostly manually chosen by the website operators. Manual selection of aspects certainly cannot scale to large number of product types as featured by general e-commerce platforms such as Amazon, Taobao and Yelp. These platforms instead turn to automatic review summarization, mined from the user review texts.

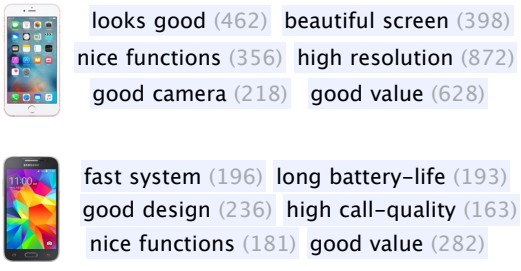

Fig. 3. Automatic review summarization for two mobile phones on an e-commerce website

An example of automatic review summarization commonly

seen on e-commerce websites is shown in Fig. 3. There is a summary of opinion phrases for each phone model, along with the frequency of each phrase mentioned in the reviews. There are two major drawbacks in this form of summarization. First, it is restricted to using the reviews about a specific product. Therefore, summaries are incompatible across different products within the same category - different models may not share the same set of opinion phrases. This makes it less useful for users to compare different models. Second, users' sentiment toward each aspect cannot be quantified and computationally compared - the difference of emotional strength between "extremely good" and "above average" is hard to capture.

Our goal in this paper is to develop an unsupervised system for aspect word extraction from a set of user reviews about products within the same category. The formulation of aspects extraction, which is extracting words from a set of documents, is similar to topic modeling where aspects can be seen as topics. However, there are three main factors that make aspect extraction more challenging:

- In topic modeling, there is no restriction on the relationship between the topics. In aspect extraction, however, we expect the final aspect words to have small semantic overlap with each other, otherwise the vagueness would confuse the consumers.
- The expression of opinions can be very versatile. In user reviews, aspects can appear both explicitly by direct reference, and implicity through users' personal experience.
- Opinions about various aspects are expressed within a short piece of text, so the topics may shift very quickly from sentence to sentence. Sentences close to each other can often talk about very different topics.

Most previously proposed unsupervised methods for aspect extraction are variations of topic models. The main problem with topic-model-based methods is they leverage only word frequencies and co-occurrances, not word semantics, so they cannot effectively leverage the sentences that implicitly discuss the product aspects. In our method we leverage the distributed representations of words and sentences. With distributed representations, the semantic similarity between two sentences can be more accurately calculated without relying too much on the lexical information. Our proposed method consists 3 clustering steps and 2 ranking phase. With a focus on aspect extraction as part of review summarization, this paper has two main contributions:

- Proposed a new framekwork of unsupervised method for aspect extraction, as an alternative to topic models.
- Acheived state-of-the-art performance in end-to-end aspect extraction in multiple domains.

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

II. Method

The review aspect extraction problem aims to extract or infer K noun words from user review texts, each word should represent a distinct aspect or feature of a type of product or service. Here K is an constant parameter for the problem. In unsupervised models for aspect extraction, the set of reviews and the number of aspects are the only inputs. 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. This allows us to apply the model to any domain with ease.

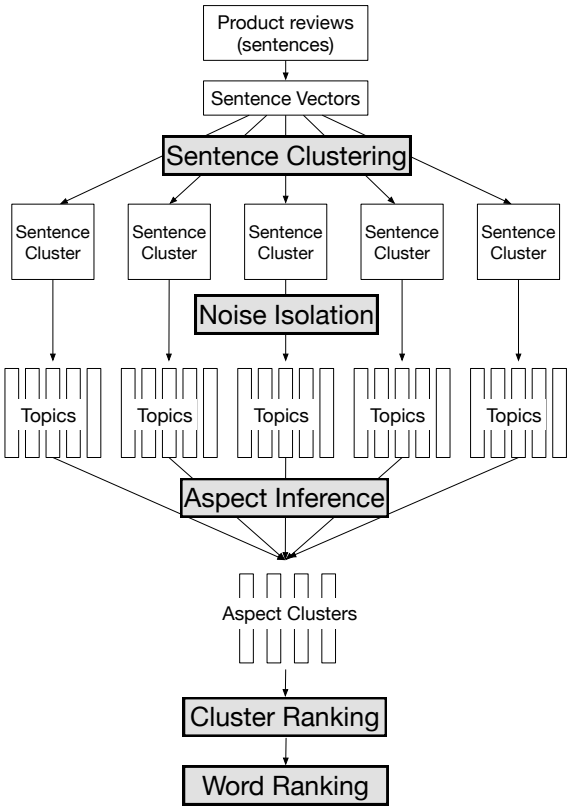

Fig. 4. Our framework.

The framework of our method is shown in Fig. 4, it consists of 5 steps:

- **Sentence Clustering**
  We convert each sentence in the reviews into a vector representation and cluster them in to N clusters of semantically similar sentences.
- **Noise Isolation**
  To isolate the noises that exist in the sentence cluster, for each sentence cluster we further generate M topics, resulting in $N \times M$ different word distributions in total.
- **Aspect Inference**
  We treat each word distribution as a vector and cluster the topics into C clusters which are the potential aspects. Here C is purposely set to be larger than K. The extra $C-K$ clusters models the redundant aspects. Each cluster contains $N \times M/C$ word distributions, or vectors. We take

the mean of these vectors to form C aspect clusters, each being a set of words and their corresponding weights.

- **Cluster Ranking**
  We define a score for the quality of each aspect cluster, and the clusters are ranked by this score.
- **Word Ranking**
  We use WordNet to calculate the semantic distances between the words in each cluster and adjust the ranking based on the both the distances and the weights of the words. The top words in each cluster are candidates of the aspect words.

In the following we will explain the motivation and the details of each of the five steps.

A. Sentence Clustering

One important feature of user reviews is that many topics are compressed into a short paragraph, where each topic corresponds to a potential aspect of the product. A typical hotel review extracted from Fig. 1 is shown as follows:

> 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.

In user reviews, topics can shift very quickly. Sentences that are close to each other may refer to completely different aspects about the product. Also, sentences about the same aspect may not appear in the review consecutively. The existence of such fine-grained semantic shifts in user reviews makes it difficult to apply the the bag-of-word abstraction of normal topic model on reviews. Therefore we propose to work on the sentence level instead of the document level, and it would be helpful if we can divide the reviews into topic-oriented segments.

Driven by this observation, in our method the first step is sentence clustering. For this purpose, we represent each sentence in a high-dimensional vector space. Instead of using simplistic methods like bag-of-word vector, we leverage a recent development of neural network in natural language processing, the distributed representation. In a distributed representation, words and sentences are converted into real-valued vectors. The distance of the vectors in the vector space will capture the semantic similarity of the words or sentences. In this work we attempt two models, recurrent neural network (RNN) and paragraph vector (PV) [2].

a) Recurrent Neural Network: To use RNN for obtaining sentence vectors, we train a neural language model on the review sentences. After the perplexity converges, we use the trained network to process each sentence of the dataset and take the last hidden vector as the vector representation of the sentence. In our method we use a variation of RNN, long-short term memory (LSTM) [3] which is reported to have better performance at modeling long sentences [4].

b) Paragraph Vector: PV is a simple but powerful extension to Word2vec [5] with two components, distributed memory (PV-DM) and distributed bag-of-word (PV-DBOW). The first one is similar to skip-gram in Word2vec and the second is similar to CBOW [5]. An important advantage of paragraph vector models is that they require no labeled data. Also, it doesn't require human experts to assign weights for words in a paragraph based on linguistic knowledge. The learned vector representations inherit an important property of Word2vec, that is the semantics similarity. Also the final paragraph vector captures the word order information with the part learned from PV-DBOW with the n-gram model. In practice, an advantage of paragraph vector over RNNs is that it can leverage trained word vectors. The word vectors can be trained on a much larger corpus so they capture the semantics relationships more accurately. Consequently, the paragraph vectors can be trained on a relatively small dataset. The performance of paragraph vectors can also be boosted by using pre-trained word vectors that are trained on a larger dataset [6]. Also, the paragraph vector models are simple and don't require the storage of a lot of information. In contrast, for RNN we need to store every state during the forward pass for back-propagation, which is very memory consuming.

c) Clustering Sentence Vectors: We run a k-means clustering on the sentence vectors and generate N sentence clusters. We then collect the sentences from the same review that are clustered together to form smaller pieces of reviews. Each review document is divided into several shorter documents, each belonging to one of the N clusters. As a result, we obtain N clusters of shorter documents.

B. Noise Isolation

The first step, sentence clustering, we might include noise and the sentences within a cluster might not all be about the same aspect. The reason is due to the common occurrences of sentences such as the following in the reviews (taken from TripAdvisor):

> 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.
>
> There is a restaurant just 5 min walk away with nice italian food, pizza was great.

In the first sentence multiple aspects are mentioned; in the second sentence, the only aspect is location however lexically it seems to be talking about food. With these complicated structures within, it is difficult for RNN or PV to correctly determine the aspects in these sentences. The result is overlaps between clusters about different aspects and noises within each cluster. To isolate the noises and resolve such overlap, We apply LDA [7] topic modeling within each sentence cluster, treating each review piece as a document, and generating M smaller topics. This will give us in total $N \times M$ topics, or, word distributions. Table I shows an example of topics inferred from three sentence clusters from hotel reviews and illustrates the

overlap problem. In this example, five topics were extracted from each sentence cluster, and each row is one topic. It can be seen that the aspects for the 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. The noise isolation step effectively separates the noise topics from other topics semantically corehent within a sentence cluster.

TABLE I
TOPICS EXTRACTED FROM THREE SENTENCE CLUSTERS OF HOTEL REVIEW.

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

C. Aspect Inference

The overlapping topics can be distinguished from other topics from the same cluster by comparing their word distributions, and in this step we resolve this overlapping and infer the candidate aspects.

We treat each topic as a vector, where an entry is the frequency of a word. These vectors have dimensionality equal the size of the vocabulary, which is too large for clustering, so we perform dimensionality reduction these vectors. Specifically, we use PCA to reduce the topics vectors to 100-dimensional. By doing this, we select the 100 most words that best distinguish different topics.

Then we perform a k-means on the $N \times M$ topics vectors to generate C clusters, each containing $(N \times M)/C$ topics. Because of the existence of noisy topics as the last topic shown in Table I, we need to set C slightly larger than the desired number of product aspects K, so that the noisy topics can be clusters together and later discarded. In an experiment we will evaluate the influence of this redundant clusters on the quality of the final aspects.

Finally, for each cluster we take the mean of the $(N \times M)/C$ topics and normalize it for the word distribution of that cluster. We call them *aspect clusters*. Some example aspect clusters extracted from hotel reviews are shown in Table II.

D. Cluster Ranking

In the previous step, we have formed C aspect clusters, with the possibilities of a few noise clusters. In this step, we discard the $C - K$ noisy clusters. To identify the noisy clusters, we design a *distinctiveness score* to measure the quality of the clusters.

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

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

The distinctiveness of a cluster measures how different it is from other clusters. Intuitively, if a cluster is similar to other clusters, it is likely to be the result of overlaps, thus it is said to have a lower quality. For the ith cluster C_i ($i \in [1, C]$), the distinctiveness score $S(i)$ is defined by:

$$\begin{aligned} S(i) &= \sum_{w \in C_i} S_i(w) \\ &= \sum_{w \in C_i} \log \left(\frac{f_i(w)}{\sum_{j \neq i} f_j(w)} \right) \\ &= \sum_{w \in C_i} \left[\log f_i(w) - \log \sum_{j \neq c} f_j(a) \right] \end{aligned} \tag{1}$$

Subsequently, the clusters are ranked in the decending order of this score and the last $C - K$ clusters are discarded. The result is shown in Table III, the discarded cluster are greyed-out.

E. Word Ranking

In Table III, we manually selected and boldfaced the the most representative words for each cluster, shown in boldface. Each of these words can act as a summary of the other words in the same cluster, and can serve as the aspect words. However, it can be seen that not all of them have the highest frequency in their clusters. In order to automatically select the best aspect words, in this ranking step we adjust the order of the word in cluster by considering both the weights and the semantics of the words.

The weights of the words are given in the aspect inference step. We keep only nouns from this step since the appropriate

aspects should be nouns. The most prominent and representative word in each cluster is assumed to be the closest to the centroid of the word cluster. To this end, 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 Word2vec to convert the words into real-valued vectors and compute the cosine similarity. Word2vec is a very popular word embedding model and has been shown to perform well on the semantic similarity tasks [8].

In the word ranking step, we not only rank the words in each cluster so that the top word is the most prominent to that cluster, we also want to make sure that such top-ranking words do not appear in multiple clusters, as it would be inappropriate to have duplicate aspects. For this purpose, we process the clusters in a sequential order, based on the ranking given by the previous step of cluster ranking. When we calculate the score for word x the ith cluster C_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 C_i is thus defined by:

$$s_i(x) = u_i(x) \sum_{y \in C_i} \hat{x} \cdot \hat{y} - \sum_{j=1}^{i-1} s_j(x), \tag{2}$$

where $\hat{x}$ is the vector representation of x; $u_i(x)$ is the weight of x in cluster C_i. 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 the next section.

III. Experiments

In this section, we present the results of three experiments. The first one is a comparison of methods for obtaining sentence vectrs. In the first step of our framework, i.e., sentence clustering, sentence representation is crucial to the performance and hence critical to overall performance of the 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 using our model.

A. Experiment 1: Comparison of Sentence Vectors

The vector representation of sentences is a crucial 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 [9]. 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.

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

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 as input the two vectors for the sentence pair given by LSTM or PV. 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 RNN and paragraph vector, 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 [10]. 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, paragraph vectors 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 words as opposed to earlier words. The performance of different sentence vectors is included in the next experiment.

B. Experiment 2: End-to-end Evaluation of Aspect Extraction

In this experiment, we evaluate the framework on the real-world application: aspect extraction from user reviews.

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 15 categories of product or service. The review content is plain English text and we do not use any labels 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

For evaluation, we ask 5 college students proficient with English to annotate ground truth aspect words for each product category. For each category, we ask them to provide 5 different words that cover the most important aspects of the corresponding product or service. The labels provided by the 5 annotators

TABLE V
DATASET SUMMARY.

Product type	Source	No. of Reviews	No. of Words
hotel	TripAdvisor	27145	210
mobile phone	Amazon	3716	136
mp3 player	Amazon	2745	128
laptop	Amazon	5471	97
tv	Amazon	1237	102
shoes	Amazon	1748	82
headphone	Amazon	1647	122
gps	Amazon	1726	72
transportation	Yelp	3077	131
restaurant	Yelp	4016	176
gym	Yelp	2481	230
shopping	Yelp	2718	123
car dealer	Yelp	2839	190
movie	Pang et al. [11]	3000	194
car	Cars.com	1074	147

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 labels for two product categories are shown in Table VI.

TABLE VI
LABELS FOR HOTEL AND SHOPPING. 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
shopping	location product service price environment product price service location ambience price food location size facility sales location food service environment price location service facility food

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 are not concerned with the number of expected aspects in this experiments. As we can see from Fig. 5, to give the smallest loss, the number of sentence clusters should be 10 or 11. Therefore, in the following experiments, we choose N to be 10.

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

Fig. 5. Selecting the number of cluster for k-means.

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 words that we compare our model agains, 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	19/25	18/25	19/25

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. 6, we show the accuracy with different redundancy $C-K$, ranging from 0 to 4, with numbers of expected aspects ranging from 3 to 7. It can be seen that 2 redundant clusters is sufficient and more redundancy can't improve the performance.

3) Models: In this section we introduce the models used in our end-to-end comparison. We compare 6 models in our experiments, LDA as a simple baseline, D-PLDA [12] as a representative for joint aspect-sentiment models, MG-LDA [13] as a representative for aspect extraction topic model, and two variations of our model Ours(LSTM) and Ours(PV). We run the 4 models on the review data of each product type sepereately. For the main experiment, the number of aspects is fixed to 5.

a) LDA: We use LDA as a simple topic model baseline for our model. 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 [12] is an LDA variation designed specifically for modeling user reivews. In D-PLDA, only opinion phrases are modeled, and the nouns and the phrases are controlled by two seperate hidden parameters where 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.

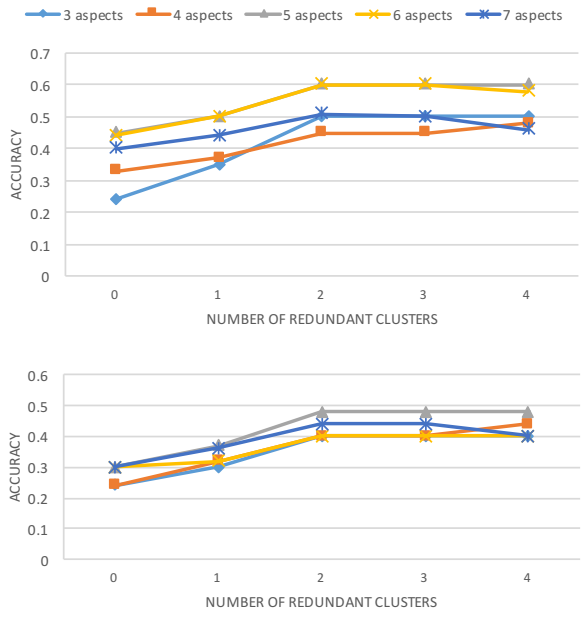

Fig. 6. The performance of different models when varying the number of redundant aspects. Top: hotel reviews. Bottom: gym reviews.

c) MG-LDA: To compare our model with a popular, well-performning model designed for aspect extraction, we choose MG-LDA [13]. MG-LDA also models topics at different grandularities. 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. 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 of a single product type.

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 Fig. 7. It can be seen that our model outperformed all other models in most of the categories (13 out of 15 with 2 ties). 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 VIII. The top words, which are chosen as the representatives of the aspects, are shown in boldface.

5) Effect of Number of Expected Aspects (K): To evaluate the effect of different numbers of expected aspects, K, we ask our annotators to provide different numbers 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. 8. It can be seen that our model performs constantly better than all other models regardless of the number of expected aspects.

TABLE VIII
TOP ASPECT WORDS FOR HOTEL REVIEWS BY DIFFERENT MODELS

Ours(PV)	**staff**, service, room, front, desk, check, concierge **food**, breakfast, bar, restaurant, coffee, morning, tea **price**, room, night, place, rate, service, money, star **room**, bed, bathroom, size, suite, floor, view, bedroom **location**, minute, square, subway, street, block, distance
Ours(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**, coffe, buffet, morning, tea, room, fruit, egg, juice **bed**, bathroom, size, floor, view, suite, king, book, decor
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

6) Effectiveness of Ranking: Here we test the effectiveness of several techiques proposed for the ranking steps. We compare three setups:

- No word ranking. Use directly the aspect cluster after step 3, 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 Fig. 9. The ranking scheme we propose, that is, work ranking with duplicate prevention, has clear advantage over the other two across all product categories.

C. Experiment 4: Aspect-based Review Summarization

In the last experiment, we demonstrate how to use our proposed aspect extraction model to construct a complete aspect-based review summarization. We use the top aspect words extracted by our model as the basis for summarization, then predict the sentiment scores using a recurrent neural network. Note that this summarization can be produced by either a set of reviews or a single reivew. In this experiment, we use our method to extract the aspects for mobile phones, and produce aspect-based review summarization for two different models of mobile phone, namely Samsung Galaxy Core Prime and Apple iPhone 6.

The aspect clusters by our method on the mobile phone review data is shown in Table IX. Since the aspects are shared across all the products within the same category, we apply our

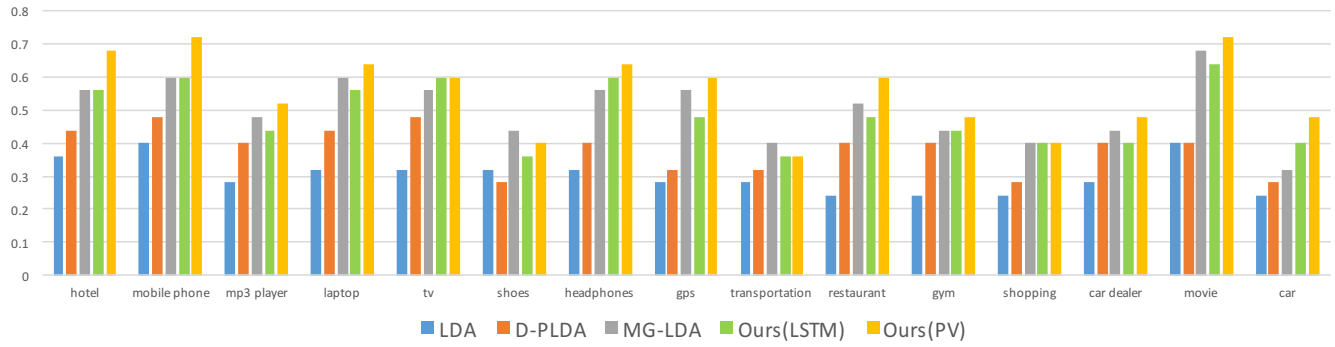

Fig. 7. Comparison of accuracies from different models on aspect extraction.

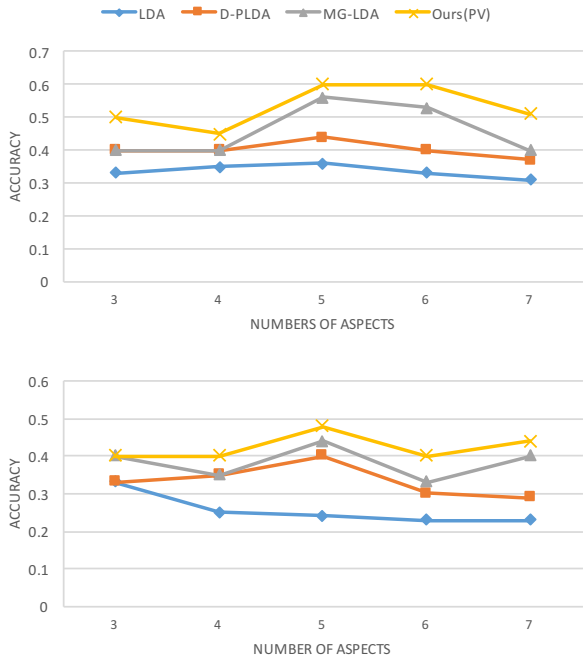

Fig. 8. The performance of different models when varying the number of expected aspects. Top: hotel reviews. Bottom: gym reviews.

TABLE IX
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

model on the whole mobile phone review dataset to extract these aspects. Then we focus on the reviews for two specific models of mobile phone and process them with a sentiment prediction backend.

We use the aspect clusters shown in the above table to identify the aspects in the review sentences by checking if the top aspect words in each cluster appeared. If so, the sentence is attached with weight equal to the highest aspect score it contains. Further, the sentiment value of the sentence is predicted with an LSTM and feed-woward network model trained on Stanford Sentiment Treebank [14]. The sentiment value for each aspect from the whole set is the weighted average of the predicted sentiment values.

The final aspect-based review summarization of two different mobile phones are shown in Fig. 10. With this summarization, users can easily compare different products with respect to the same set of important aspects, without having to read a lot of reviews.

One of the advantages of using our method for review summarization is that the chosen aspects reflect what the consumers care most about each product type. The fundamental reason behind this is that our model can truly leverage the large amount of data compared to traditional methods. In Fig. 3, we can see phrases such as “beautiful screen” and “good camera”, but we also see “high resolution”, which is confusing because we cannot know if it means the screen or the camera, and either way there is overlap in meaning. In our method, the topics of the sentence is implicitly embedded in the sentence vector in the first clustering process, so we can leverage not only the opinion phrases but other surrounding words to decide whether “high resolution” refers to the screen or the camera. For example, if the review sentence is “it has high resolution and the texts look so crisp”, we would know it’s about the screen. This allows our summarization to leverage a lot more data than these traditional methods.

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

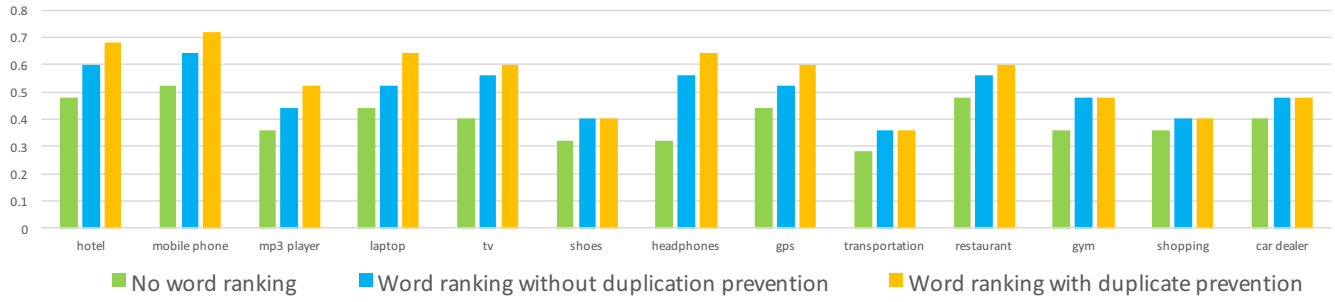

Fig. 9. The accuracy performance with different ranking setups.

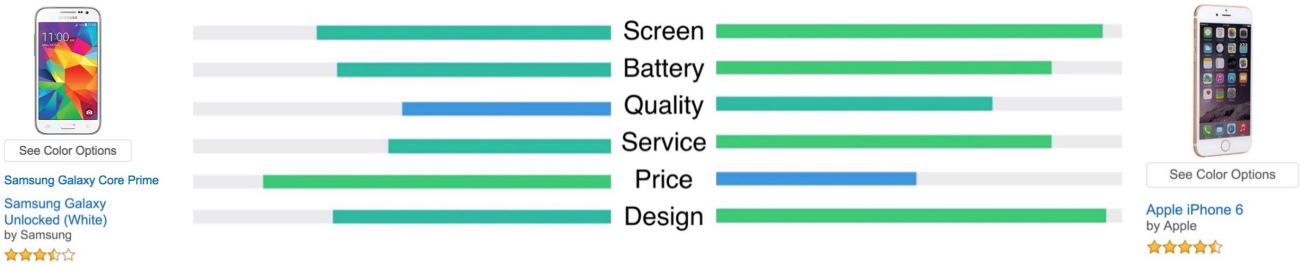

Fig. 10. Comparing two mobile phones.

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 [15] uses manually crafted mining rules. Qiu et al. 2011 [16] also used rules, plus the Double Propagation method to better relate sentiment to aspects. Gindl et al. 2013 [17] also used the Double Propagation method, with the help of anaphora resolution for identifying co-references to improve the accuracy. Su et al. 2008 [18] 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. [19] 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[20] and LDA[7]. 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 [21] models in parallel aspects and sentiments per review. Lin et al. 2009 [22] models the dependency between the latent aspects and ratings. Moghaddam et al. 2012 [12] 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 [13], 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 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 [23], [14], [24], [25] 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 to solve the aspect extraction from user reivews, which is useful for structured and quantitative review summarization and opinion mining for any product or service. Topic modeling alone doesn't do very well with this problem because reviews are highly condense and tend to switch topics quickly within themselves. Instead, we proposed a multistage framework, which clusters the reviews from coarse-grained sentences to fine-grained words. This is a purely unsupervised method, which doesn't require any human annotation. Results shows that this approach outperforms other models, including MG-LDA, which is a popular method designed specifically for aspect extraction, on 15 different categories of products and services. Although RNN and LSTM has been used widely in various sentiment analysis tasks like sentiment prediction, we find out that paragraph vector can be more effective when it comes to learning a representation under unsupervised settings, although paragraph vector is simpler than RNN and has fewer parameters.

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 *International Conference on Machine Learning*, 2014.

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

[4] R. Jozefowicz, W. Zaremba, and I. Sutskever, "An empirical exploration of recurrent network architectures," in *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, 2015, pp. 2342–2350.

[5] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in Neural Information Processing Systems*, 2013, pp. 3111–3119.

[6] T. Mikolov, W.-t. Yih, and G. Zweig, "Linguistic regularities in continuous space word representations." in *HLT-NAACL*, 2013, pp. 746–751.

[7] 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.

[8] 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.

[9] 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.

[10] 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

[11] B. Pang, L. Lee, and S. Vaithyanathan, "Thumbs up?: Sentiment classification using machine learning techniques," in *Proceedings of the ACL-02 Conference on Empirical Methods in Natural Language Processing-Volume 10*. Association for Computational Linguistics, 2002, pp. 79–86.

[12] S. Moghaddam and M. Ester, "On the design of lda models for aspect-based opinion mining," in *Proceedings of the 21st ACM International Conference on Information and Knowledge Management*. ACM, 2012, pp. 803–812.

[13] I. Titov and R. McDonald, "Modeling online reviews with multi-grain topic models," in *Proceedings of the 17th international Conference on World Wide Web*. ACM, 2008, pp. 111–120.

[14] 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 *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, vol. 1631. Citeseer, 2013, p. 1642.

[15] 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.

[16] 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.

[17] 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.

[18] 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.

[19] 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.

[20] 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.

[21] 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.

[22] 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.

[23] 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.

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

[25] 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.