Li SK, Guan Z, Tang LY *et al.* Exploiting consumer reviews for product feature ranking. JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY 27(3): 635–649 May 2012. DOI 10.1007/s11390-012-1250-z

Exploiting Consumer Reviews for Product Feature Ranking

Su-Ke Li[1] (李素科), Zhi Guan[2] (关　志), Li-Yong Tang[2] (唐礼勇), and Zhong Chen[2,*] (陈　钟), *Member*, *CCF*, *IEEE*

[1] *School of Software and Microelectronics, Peking University, Beijing 100871, China*

[2] *School of Electronics Engineering and Computer Science, Peking University, Beijing 100871, China*

E-mail: {lisuke, guanzhi, tly, chen}@infosec.pku.edu.cn

Received August 31, 2011; revised January 19, 2012.

Abstract Web 2.0 technology leads Web users to publish a large number of consumer reviews about products and services on various websites. Major product features extracted from consumer reviews may let product providers find what features are mostly cared by consumers, and also may help potential consumers to make purchasing decisions. In this work, we propose a linear regression with rules-based approach to ranking product features according to their importance. Empirical experiments show our approach is effective and promising. We also demonstrate two applications using our proposed approach. The first application decomposes overall ratings of products into product feature ratings. And the second application seeks to generate consumer surveys automatically.

Keywords product feature ranking, product review, opinion mining

1 Introduction

Recent years Web 2.0 technology leads Web users to publish a large number of consumer reviews about products and services. We can find consumer reviews in blogs, Web forums, social networks, and commercial websites such as Amazon① and Tripadvisor②. These reviews reflect consumers' expectation for products and services, and also show how consumers evaluate product or service quality. Mining knowledge from consumer reviews has attracted extensive research efforts.

In a review, a consumer may mention one or several different product features which are cared by consumers. However, not all product features are important enough to affect other consumers making purchasing decisions. A product may also have special features that are more important than other features. Therefore, to find and rank product features in a product domain is a research problem. Intuitively, if a product feature is mentioned in more reviews in the same product domain, the feature may be more important.

This work is an extended version of our previous work[1]. [1] proposes a feature ranking approach, namely Double Propagation-Based Linear Regression (DPLR), which was used to rank product features according to their importance. Actually, before [1] our earlier research work[2-3] has addressed the problem of service aspect ranking. [1] focuses on both service aspect ranking and product feature ranking. However, in [1] we did not conduct experiments to compare DPLR with the state-of-the-art product feature ranking method DP-HITS[4]③. In this work, based on our previous product feature ranking approach DPLR[1], we propose a new product feature ranking method DPLR-R (Double Propagation-Based Linear Regression with Rules). We also give some experimental comparisons between Zhang *et al.*'s method[4] and DPLR[1]. Empirical experiments show our new product feature ranking method DPLR-R is effective, and it can even rival the state-of-the-art product feature ranking method DP-HITS[4] on our datasets. We consider the product feature ranking problem as a regression problem, and we compute the weights of product features using gradient descent algorithm which can be used to get a local optimal solution.

The remainder of this paper is organized as follows. We present our product feature ranking scheme in Section 2. In Section 3, we show our experiments that

Regular Paper

This work is supported by the National Natural Science Foundation of China under Grant No. 61170263.

*Corresponding Author

①http://www.amazon.com.

②http://www.tripadvisor.com.

③Zhang *et al.*[4] did not name their approach. We name their product feature ranking approach[4] as DP-HITS.

have been conducted on the extracted Web reviews. Section 4 demonstrates two applications based on our proposed two ranking schemes: DPLR and DPLR-R. Section 5 summarizes some related research. Section 6 gives some discussions about our work. Finally, in Section 7 we conclude our work.

2 Proposed Approach: Double Propagation-Based Linear Regression with Rules (DPLR-R)

Our approach has two stages. At the first stage, we extract product features using a hybrid method that is based on double propagation (DP) algorithm[5] and two extraction rules, and we also extract opinion units using the two extraction rules. Then we rank product features using our linear regression method. We name our method Double Propagation-Based Linear Regression with Rules (DPLR-R).

2.1 Product Feature Extraction

We firstly need to extract product features. The extraction process has two steps as follows.

• In the first step, we use double propagation (DP) algorithm[5] to extract candidate product features. Let the extracted product feature set be A.

• We also use two rules to extract candidate product features and opinion units. Let the extracted product feature set be B.

We get the final candidate product feature set $F = \{f|f \in C, C = A \bigcap B, support(f) > \eta\}$. The constant value η is the support value of the candidate product features extracted from the dataset. The support value means the proportion of reviews which mention the candidate features. For example, in this work η is 0.000 5 that means we only extract candidate product features which are mentioned in at least 5 consumer reviews if the total number of consumer reviews is 10 000.

Product features can be product components or product attributes[6]. We use nouns and consecutive nouns as candidate product features or service aspects. Sometimes, a product feature is not a single noun, but several consecutive nouns. For instance, in the sentence "When paired with a compatible 3G network, you'll enjoy a high-speed connection offering a variety of feature-rich wireless services–from data connectivity to your office to multimedia streaming...", "data connectivity" is a product feature for mobile phone, however, "connectivity" can be a product feature too if we can infer that the opinion for connectivity is about data connectivity in the language context. In order to express concisely, we do not distinguish product features or service aspects, and we deem all of them as product features.

Double propagation (DP)[5] exploits the relations between sentiment words and product features that the sentiment words modify to extract product features. Double propagation assumes that product features have high probability to be associated with sentiment words, and sentiment words may co-occur with product features. The basic idea of double propagation is to use extracted feature words to extract sentiment words, then we can use extracted sentiment words to extract latent features. The process looks like double information propagation. Double propagation algorithm converges when the feature word set and the sentiment word set do not change any more.

Double propagation needs initial sentiment seed word set to bootstrap the double propagation process. In this work, double propagation is independent of review ratings, and it relies on dependency trees generated by Minipar④. As a state-of-the-art rule-based feature extraction method, double propagation depends on two dependence relations: direct relation (DR) and indirect relation (IDR). The two relations are expressed as eight extraction rules. For more detailed information, readers can refer to Qiu *et al.*'s research work[5].

We use double propagation method and two rules to extract product features. To use double propagation, we also use Minipar to generate dependence trees. We adopt the initial subjectivity lexicon[7] as DP's initial seed set. After we have extracted candidate product feature set A, we also use two rules to extract the other candidate product feature set B.

In this work, noun units are extracted as candidate product features, and opinion units are extracted also. In order to express clearly, we give two definitions as follows.

Definition 1 (Noun Unit). *A noun unit is a unit which contains maximum consecutive nouns in a sentence clause. For instance, both "breakfast" and "breakfast room" are noun units.*

Definition 2 (Opinion Unit). *An opinion unit is an opinion word or a combination of an opinion word and a negative indicator.*

Tokens such as "not", "no", "donot", "don't", "didnt", "didn't", "wasnt", "wasn't", "isnt", "isn't", "weren't", "werent", "doesnt", "doesn't", "hardly", "never", "neither", and "nor" are negative indicators. If a negative indicator is in the left context window of an opinion word, the combination of negative indicator and the opinion word is called opinion unit. Because the reviews were written freely by Web users, some expressions are not formal, such as "didnt" means "did

④http://www.cs.ualberta.ca/∼lindek/minipar.htm.

not". If a negative indicator is in the left context window of an opinion word, then the negative indicator and the opinion word are concatenated together to form an opinion unit. These negative indicators may be adjacent to product opinions or not. For example, there are two opinion units: "isn't friendly" and "isn't helpful" in the sentence "The manager isn't friendly and helpful". The negative indicator "isn't" is in the left context window of "helpful" and "friendly". The distance between "isn't" and "friendly" is 1, and it is in the left context window $[-3, 0]$ of "friendly". The distance between "isn't" and "helpful" is 3, and it is also in the left context window $[-3, 0]$ of "helpful". Besides, the opinion words "friendly" and "helpful" are also opinion units if there are no negative indicators around their window contexts.

After we get candidate product feature set A using double propagation algorithm, we need to get product feature set B. In order to extract candidate product feature set B and opinion units using our rules, we carry out some necessary preprocessing. We segment all product reviews into sentences, and we get all POS (Part-of-Speech) tags for these sentences. For each sentence with POS tags, we then extract candidate product features and opinion units using following two rules:

• *Rule*-1. If a noun unit has one or several adjective modifiers, each adjective modifier is looked as a candidate opinion unit and the consecutive nouns together are looked as a candidate product feature. The adjective modifier must have POS tag of JJ, JJR, or JJS. And the candidate product feature has POS tag of NN or NNS. This rule is based on the observation that consumers like to use adjective modifiers to describe product features. For instance, in the noun phrase "great screen", "great" is an opinion unit, while "screen" is a candidate product feature.

• *Rule*-2. In a sentence clause, if the distance between an adjective (with POS tag of JJ, JJR, or JJS) and a noun unit is less than d ($d = 4$ in our work), the adjective is also in the right side of the noun unit, and there is a word tagged with POS tag of VB, VBG, or VBP between the adjective and the noun unit, then we extract the noun unit as candidate product feature and the adjective and its associated negative indicator as an opinion unit.

The above two rules can extract candidate product feature set B and candidate opinion units. Remember, in our approach we combine double propagation algorithm and two rules to extract candidate product features. After we get candidate product feature set A using double propagation algorithm, and we get the candidate product feature set B using our two rules, as we have stated in the beginning of this subsection, the final candidate product feature set is $F = \{f | f \in C, C = A \bigcap B, support(f) > \eta\}$.

2.2 Product Feature Ranking

Web users publish their opinions on product features or service aspects. We assume a consumer review only focuses on one product, and the publisher may mention several product features in each product review. We represent a consumer review as a feature-opinion vector $\boldsymbol{review}_z$.

$$\boldsymbol{review}_z = \langle (f_1, O_{z_1}), (f_2, O_{z_2}), \ldots, (f_n, O_{z_n}) \rangle,$$

where in each feature-opinion pair (f_j, O_{z_j}), f_j is the product feature, and O_{z_j} is an opinion unit set associated with product feature f_i in $\boldsymbol{review}_z$. For each product features, there may be one or several opinion units associated with it.

Consumers give their subjective product quality evaluation through publishing reviews. Some websites also allow users to give over ratings for products and services, such as Amazon and Tripadvisor. These overall ratings reflect the quality evaluation for products. For different product features concerned with a consumer, there are different weights associated with them. That is to say, from a consumer's point of view, some product features are important, but some are not. Let $\boldsymbol{review}_z$ be the z-th review, and r_z is original overall rating associated with $\boldsymbol{review}_z$. The rating r_z was given by consumers. We can get predicted rating r_{z_p} for $\boldsymbol{review}_z$. An overall predicted rating r_{z_p} can be expressed by overall ratings of product features mentioned in $\boldsymbol{review}_z$.

$$r_{z_p} = \sum_{1 \leqslant i \leqslant n_z} w_i r_{z,f_i}, \tag{1}$$

where r_{z_p} is the predicted overall rating associated with $\boldsymbol{review}_z$, w_i is the weight of product feature f_i, n_z is the number of features mentioned in $\boldsymbol{review}_z$, and r_{z,f_i} is the rating of product feature f_i in $\boldsymbol{review}_z$. Therefore, if we know the feature ratings, we can estimate overall rating for a product. Because in a review, a product feature may be associated with several opinion units, we employ ratings of opinion units associated with the feature to estimate the rating of the feature.

In order to get the rating r_{z,f_i} for a product feature f_i mentioned in the consumer review $\boldsymbol{review}_z$, we need to get the overall ratings for opinion units

$$r_{o_j} = \frac{\sum_{1 \leqslant z \leqslant |R_{o_j}|} r_z}{|R_{o_j}|}, \tag{2}$$

where r_z is the original rating associated with a review which contains opinion unit o_j, and R_{o_j} is the set of all the reviews that contain o_j. Then we can get the rating for product feature f_i in $\boldsymbol{review}_z$,

$$r_{z,f_i} = \frac{\sum_{1 \leqslant j \leqslant |O_{z_i}|} r_{o_j}}{|O_{z_i}|}, \tag{3}$$

where r_{z,f_i} is the rating for feature f_i in a review, and r_{o_j} is the overall rating for opinion unit r_{o_j} (gotten by (2)), and O_{z_i} is opinion unit set associated with the product feature f_i in $\boldsymbol{review}_z$.

Suppose we have review set R, and the size of the dataset is n ($n = |R|$). Here r_{z_p} ($1 \leqslant z \leqslant n$) is the predicted rating for $\boldsymbol{review}_z$ calculated by using (1). If we can extract total m kinds of product features from R, then $\boldsymbol{w}$ is the weight vector for all features extracted from the reviews, and

$$\boldsymbol{w} = (w_1, w_2, w_3, \ldots, w_m).$$

We use (4) to get the weights of product features

$$\boldsymbol{w} = \arg\min_{\boldsymbol{w}}((1-\alpha) \times \frac{1}{n} \sum_{1 \leqslant z \leqslant n} L(r_{z_p}, r_z) + \alpha \times \lambda \|\boldsymbol{w}\|^2), \tag{4}$$

where n is the number of reviews, $L(r_{z_p}, r_z) = (r_{z_p} - r_z)^2$, r_{z_p} is the predicted rating and r_z is the original rating associated with the review $\boldsymbol{review}_z$. If we can extract m distinct product features from review set R, and we can extract l distinct opinion units from review set R, and then for all the reviews in review set R, for every review $\boldsymbol{review}_z$ we construct $m \times l$ feature-opinion matrix $\boldsymbol{M}_z$. In this case, if a feature-opinion pair is extracted from $\boldsymbol{review}_z$, then $\boldsymbol{M}_{z_{x,y}} = 1$, otherwise $\boldsymbol{M}_{z_{x,y}} = 0$, where x is the feature dimension number and y is the opinion dimension number. Let $\boldsymbol{O}_z$ be the column vector for opinion units mentioned in $\boldsymbol{review}_z$, $\boldsymbol{O}_z = \langle o_{z_1}, o_{z_2}, \ldots, o_{z_l} \rangle^{\mathrm{T}}$, where T denotes transpose of the vector $\langle o_{z_1}, o_{z_2}, \ldots, o_{z_l} \rangle$, if an opinion unit mentioned in $\boldsymbol{review}_z$, then the corresponding element of vector $\boldsymbol{O}_z$ is the overall rating of the opinion unit, otherwise, the element value is 0. Therefore we can predict the overall rating for review r_z with $r_{z_p} = \boldsymbol{w}\boldsymbol{M}_z\boldsymbol{O}_z$. If $\alpha = \frac{1}{2}$, then (4) can be expressed as

$$\boldsymbol{w} = \arg\min_{\boldsymbol{w}} \Big(\frac{1}{2n} \sum_{1 \leqslant z \leqslant n} (\boldsymbol{w}\boldsymbol{M}_z\boldsymbol{O}_z - r_z)^2 + \frac{1}{2}\lambda \|\boldsymbol{w}\|^2 \Big). \tag{5}$$

Although we can get the closed-form solution to (5), to get the final result for the solution we may need calculation such as matrix inverse carried on matrixes that are not sparse and very huge. In order to make the computation and implementation easy, we use gradient descent algorithm GetFeatureWeights to get the feature weight vector. Let $L(\boldsymbol{w}) = (\frac{1}{2n} \sum_{1 \leqslant z \leqslant n} (\boldsymbol{w}\boldsymbol{M}_z\boldsymbol{O}_z - r_z)^2 + \frac{1}{2}\lambda \|\boldsymbol{w}\|^2)$, to get vector $\boldsymbol{w}$, we turn to use some numerical method to get gradient of $\boldsymbol{w}$ (denoted as ∇).

We use gradient descent algorithm GetFeatureWeights to get the feature weight vector, as Algorithm 1 shows. In the Algorithm GetFeatureWeights, the value of each element of the feature weight vector $\boldsymbol{w}$ is initialized to 0.1. Now we give some analysis about computational complexity of algorithm GetFeatureWeights. If we can extract m product features and l opinion units from n reviews, then the time complexity of computing (6) is $O(nml)$. We can also know the time complexity of computing (5) is $O(nml)$ too. Suppose the $maxIterNum$ in algorithm GetFeatureWeights is N, and the inner iteration number of **while** is M, then the computational complexity of algorithm GetFeatureWeights is $O(NMnml)$.

$$\begin{aligned} \nabla_{\boldsymbol{w}} L &= \frac{\partial L(\boldsymbol{w})}{\partial \boldsymbol{w}} \\ &= \frac{1}{n} \sum_{1 \leqslant i \leqslant n} (\boldsymbol{w}\boldsymbol{M}_z\boldsymbol{O}_z - r_i)(\boldsymbol{M}_z\boldsymbol{O}_z)^{\mathrm{T}} + \lambda \boldsymbol{w}. \end{aligned} \tag{6}$$

Algorithm 1. GetFeatureWeights

Input: For n reviews, feature-opinion $m \times l$ matrixes $\boldsymbol{M}_1, \boldsymbol{M}_2, \ldots, \boldsymbol{M}_n$;
For n reviews, opinion vector $\boldsymbol{O}_1, \boldsymbol{O}_2, \ldots, \boldsymbol{O}_n$;

Output: Feature weight vector $\boldsymbol{w}$;

```
iterNum = 1;
maxIterNum = n;
while iterNum < maxIterNum do
    Get gradient ∇_w L using (6);
    γ = 1;
    while (L(w_t − γ∇_w L) ⩾ L(w_t)) do
        γ = γ × 0.5;
    end while
    w_{t+1} = w_t − γ∇_w L;
    iterNum = iterNum + 1;
    if L(w_t) − L(w_{t+1}) < ε then
        Break;
    end if
end while
return w;
```

3 Experiments

3.1 Datasets and Setting

We crawled Web pages from Amazon⑤ and Tripadvisor⑥, and we extracted total 6 000 phone, laptop and hotel related consumer reviews. We used

⑤http://www.amazon.com.

⑥http://www.tripadvisor.com.

Sentence Detector API of OpenNLP[⑦] to get all sentences from the reviews. We used Part-of-Speech (POS) Tagger API to get Part-of-Speech tags for each sentences. We used Chunking API of OpenNLP to get chunks for each sentences. The statistics of our datasets are shown in Table 1.

Table 1. Statistics of Our Dataset

Product Domain	No. Positive Reviews	No. Negative Reviews	No. Sentences
Phone	1 000	1 000	28 111
Laptop	1 000	1 000	14 816
Hotel	1 000	1 000	18 694

We also need to set some parameters for regression. In Algorithm GetFeatureWeights, the max iteration number n is 20, and $\epsilon = 0.05$. And in (4), we set $\alpha = \frac{1}{2}$ and $\lambda = 0.5$.

In Algorithm GetFeatureWeights, we use L2-norm regularization (see (4)) to get product feature weights. Actually we also can using L1-norm regularization to get the product feature weights too. L2-norm regularization is easy to use and it does not let the ranking problem more complexity compared with L1-norm regularization. L2-norm regularization problems are differentiable, so we can get the related derivative more easily using gradient decent method. Although L1-norm regularization could yield sparse models that are more easily interpreted[8], L1-norm regularization problems are not differentiable. Hence, if we want to solve L1-regularization problem we need other ways such as convex optimization and iterative methods. In order to compute product feature ranking weights more quickly and conveniently, we choose L2-norm regularization method in our ranking approach DPLR-R.

In our experiments we found that if we used all product features in the regression, the ranking performance was not good. Hence, we do not use all the candidate product features to conduct linear regression. For DPLR-R, we remove the low frequency candidate product features with support value 0.000 5 (using product features with support value that is greater than 0.000 5). For DPLR method, the parameters are set as our previous work[1].

3.2 Comparisons with Other Methods

We have compared our approach with several other methods that can be used to rank product features. They are Double Propagation Frequency (DPF) (baseline), FORank, Linear Regression with Context Windows (LRCW), Double Propagation-Based Linear Regression (DPLR), and DP-HITS[4]. Here we briefly describe these methods respectively.

3.2.1 Double Propagation Frequency

Double Propagation Frequency (DPF) is based on double propagation algorithm[5] described in Subsection 2.1. DPF method ranks candidate product features which are extracted using double propagation algorithm according to their frequencies appearing in our corresponding product dataset. DPF is also used as the baseline to rank product features in the research work[4]. DPF is our baseline method.

3.2.2 FORank

Using this method, we consider the product feature ranking problem as the problem of finding the top K most influential nodes in a feature-opinion graph. Using link structure information, PageRank[9] is a famous ranking algorithm to predict the importance of Web pages. FORank (feature-opinion rank) is based on Pagerank. The rationale of generating edges is based on our hypothesis: *If there are more kinds of sentiment words associated with a product feature, the product feature has high probability to be an essential product feature. If there are more kinds of product features associated with a sentiment word, the sentiment word has high probability to be an important opinion word.* In our previous work[2], we have ranked service aspect using PageRank-based algorithm AORank. We construct the directed graph G using opinion words and product features. We have two rules to extract feature-opinion pairs.

• Extract adjective modifiers as opinion word candidates and nouns or consecutive nouns in noun phrase as product feature candidates.

• Extract any nouns from a noun phrase as feature candidates and all adjectives from an adjective phrase as opinion word candidate, where the noun phrases and the adjective phrases are in the chunk sequence of ⟨NP (Nonn Phrase) VP (Verb Phrase) ADJP (Adjective Phase)⟩ that is generated by a POS tagger.

Because of the limitation of POS tagging method, some words are classified into both adjective class and noun class. In order to distinguish these words, we assign a word into its class in which it has higher frequency between two classes. In the graph G, product features and opinion words are two kinds of nodes. G is a bipartite graph.

If a sentiment word co-occurs with a product feature word, then there are two direct edges are generated: one from the product feature to the opinion word, the other is from the opinion word to the product feature. In this work, if an opinion word is in the context window of a product feature or vice verse, the opinion word is associated with the product feature.

⑦http://www.opennlp.org.

We consider that ranking features and opinion words is a PageRank's random surfer process[9]. Matrix $\boldsymbol{A}$ is the adjacency matrix or stochastic transition matrix of G which has n nodes. Let $\boldsymbol{P}$ be an n-dimensional column vector of aspect-opinion rank values:

$$\boldsymbol{P} = (p_1, p_2, \ldots, p_n)^{\mathrm{T}}. \tag{7}$$

Let S be the product feature node set and O is the opinion word node set, then $n = |S| + |O|$. Matrix $\boldsymbol{A}$ is the adjacency matrix with

$$\boldsymbol{A}_{ij} = \begin{cases} \dfrac{1}{O_i}, & \text{if } (i,j) \in \boldsymbol{E}, \\ 0, & \text{otherwise}, \end{cases} \tag{8}$$

where O_i is the out-degree of node i and E is an $n \times n$ matrix of all 1's. Node i can be a product feature or an opinion word. The PORank model is

$$\boldsymbol{P} = ((1-d)\frac{\boldsymbol{E}}{n} + d\boldsymbol{A}^{\mathrm{T}})\boldsymbol{P}, \tag{9}$$

where $d \in (0,1)$, d is a damping factor. We also use power iteration method[10] to compute the final rank vector $\boldsymbol{P}$ until it converges.

In our experiments, the maximum number of iteration is 20. The iteration algorithm is as Algorithm 2 shows. In Algorithm 2, ε is a pre-specified threshold.

Algorithm 2. Aspect-Opinion-Iterate
Input: Feature-opinion Graph G;
Output: Ranked feature-opinion column Vector $\boldsymbol{P}$;

```
1    //e is a column vector of all 1's.
2:   P_0 ← e/n;
3:   j ← 1;
4:   IterNum ← 0;
5:   while True do
6:       P_j ← (1 − d)e + dA^T P_{j−1};
7:         //ε is a pre-specified threshold.
8:       if ||P_j − P_{j−1}|| < ε or IterNum > MaxIterNum
9:         then break;
10:      end if
11       IterNum = IterNum + 1
12:      j ← j + 1;
13:  end while
14:  return P_j;
```

3.2.3 Linear Regression with Context Windows

Linear Regression with Context Windows (LRCW) is independent to dependence trees, and it only extracts opinion units and product features in the context windows of opinion words. The extraction method is straightforward. A product feature can be a single noun or several consecutive nouns. For instance, in the sentence "The minibar was ridiculous, the breakfast was ridiculous and even the room service was totally awful!", "room service" is a service feature (or it can be a "service aspect"). Although "service" can be an independent feature and a general concept, "room service" shows the service is room service, not about other services. Therefore we classify "room service" as a feature. The further consideration to product feature extraction is about how to construct hierarchical concept structure of product features belonging to the same kind of product.

In this method, we use adjectives as candidate opinion units since adjectives can be used in product mining for candidate opinion words[11]. We only consider noun and several consecutive nouns around adjectives as candidate product features, so we get the second-order context window around each candidate opinion word in reviews. For example, $[-3, +3]$ is the context window around the candidate opinion word in the same sentence. In the sentence "Hotel owner is nice and helpful", opinion word *nice*'s $[-3, +3]$ context window contains "hotel owner". We extract words with POS tags of JJ, JJR, JJS as candidate opinion words, and words with POS tags of NN, NNS as candidate product feature. We also extract negative indicators which can form opinion units if they appear in the context windows of opinion words.

LRCW is different from DPLR. DPLR uses double propagation algorithm to extract product features and opinion units but LRCW does not. However, the experimental results show the performance of LRCW approximates to the performances of DPLR. We do not use all the candidate features to conduct linear regression. For DPLR, we remove the low frequency candidate product features using support value 0.005. It means if the percent of reviews containing the candidate feature is lower than 0.5%, then the candidate feature is not used for LRCW.

3.2.4 Double Propagation-Based Linear Regression

Double Propagation-Based Linear Regression (DPLR) is also based on double propagation algorithm. The ranking process is almost the same as DPLR-R. DPLR was proposed in our previous work[1]. But DPLR does not use rules to extract candidate product features and opinion units. The differences between DPLR and DPLR-R lie in twofold.

- DPLR-R uses both double propagation algorithm and rules to extract product features and opinion units.
- DPLR-R uses opinion units extracted by rules to do linear regression, however, opinion units for DPLR are in the context windows of noun units which are

looked as candidate product features.

In this case, an opinion unit is associated with a product feature if the opinion unit contains at least one sentiment word extracted by using DP algorithm and it is also in the context window of the product feature in the same sentence clause. The context window is $[-3, 3]$ which means we only consider opinion units that have maximum distance value of 3 between any token of opinion units and product features.

3.2.5 Double Propagation-Based HITS

DP-HITS (Double Propagation-Based HITS) approach was proposed by Zhang *et al.*[4] DP-HITS also adopts double propagation algorithm to extract product features. To deal with product feature ranking problem, DP-HITS ranks product features by two factors that are feature relevance and feature frequency. Feature relevance is showed as a bipartite directed graph which is constructed by dependency relations and part-whole relation. The dependency relations include direct relations and indirected relations which are determined by double propagation algorithm. After getting all these relations among opinion words and candidate product features, the Web page ranking algorithm HITS[12] is applied to rank and find the important product features.

Zhang *et al.* gave the final ranking function that is given in (10).

$$S = S(f) \log(freq(f)), \tag{10}$$

where $freq(f)$ is the frequency count of candidate product feature f, and $S(f)$ is the authority score of f that is computed by HITS algorithm.

3.3 Evaluation Metric

MAP (mean average precision) is often used in evaluation of information retrieval.

$$\text{MAP} = \frac{\sum_{r=1}^{N}(P(r) \times rel(r))}{r_n}, \tag{11}$$

where r is the rank, N is the number of candidate product features, r_n is the number of product features identified by human, $rel(\cdot)$ is a binary function on the relevance of a given rank, and $P(r)$ is precision at rank r.

We use MAP to evaluate results of product feature ranking. We employ two students to be judges to determine whether candidate product features are true product features. The two students know their task and our definition of product feature clearly. For a candidate product feature, if any judge deems the candidate as a product feature, then the candidate is classified as a true product feature; otherwise, if both judges do not regard the candidate as a product feature, the candidate is not classified as a true product feature.

3.4 Experimental Results

Some nouns bear category information. For example in our hotel dataset, "hotel" is a category concept, but we do not classify "hotel" as a service feature. The same rule is applied to our phone dataset and laptop computer dataset: "phone" and "laptop" are not a product feature. Judging what candidate features (noun or consecutive nouns) are product features is subjective. The thumb-rule is if a candidate feature is a component of a product, or the candidate feature is an attribute of a product, then the candidate feature is a product feature. This is consist with product feature definition given by Liu[6]. It is common that a candidate feature is a product feature in one product domain, but it is not product feature in the other product domains.

Table 2 gives the top 20 ranked candidate product features extracted from the phone dataset using different methods. The interesting results of LRCW shows that phone consumers are likely to mention "camera", "quality", "reception", "keyboard", "screen", and "touch screen" in their product reviews. For DPLR, we get that consumers are mostly concerned with product features such as "camera", "screen", "reception", "quality" and "keyboard". DPLR-R gives phone's top 5 most important features which are "quality", "camera", "screen", "keyboard", and "use". "Time" and "thing" are not product features. DP-HITS shows the top 5 most important features are "screen", "feature", "battery", "quality", and "service".

Table 2. Top 20 Features of the Phone Dataset

Approaches	Top 20 Features
DPF	phone, screen, battery, time, camera, quality, keyboard, life, problem, call, service, text, day, music, button, card, thing, way, cell, data, touch
FORank	phone, screen, thing, time, phones, camera, bit, keyboard, quality, feature, use, features, problem, problems, people, key, battery, standard, buttons, message, user
LRCW	phone, camera, quality, reception, keyboard, screen, touch screen, improvement, life, service, texting, nothing, things, cell phone, picture quality, size, typing, plan, call quality, battery
DPLR	phone, camera, screen, reception, quality, keyboard, life, purchase, size, battery, texting, user, service, feel, sound, music, store, internet, speakerphone, memory, light
DP-HITS	phone, screen, feature, battery, time, quality, service, use, device, thing, problem, keyboard, camera, one, way, button, call, life, cell, software
DPLR-R	phone, quality, camera, screen, keyboard, time, thing, use, battery, reception, memory, look, device, sound, size, access, apps, problem, choice, way

For the top 20 phone product features ranking, DPLR seems to have better performance than other methods. It is interesting that feature "price" is not in the top 20 list. It seems that consumers care more about multi-media functionalities provided by phones. And "camera", "screen" are two most important product features of phones. Phone consumers look likely to care more about "quality" and "service" about phones than "price" too.

Table 3 is the top 20 product feature ranking results for laptop. DPLR-R shows the five most important features including "price", "mouse", "quality", "bag", and "size". DP-HITS has "backpack", "zipper", "compartment", "bag", and "mouse" as the most extract product features. Compared to DP-HITS, the five top features of DPLR-R are more reasonable. DPLR shows "laptop", "price", "mouse", "bag", "quality", "screen" are the most important features for laptop computers. We also find some candidate features such as "fit" describing laptops are easily to be carried. However, "fit" is not a product feature.

Table 3. Top 20 Features of the Laptop Dataset

Approaches	Top 20 Features
DPF	laptop, computer, battery, product, bag, time, mouse, screen, price, problem, case, thing, netbook, power, quality, sleeve, pad, keyboard, drive, money, way
FORank	laptop, thing, sound, computer, bit, screen, software, design, bag, problem, case, product, mouse, plastic, machine, surface, quality, time, color, drive, price
LRCW	laptop, price, product, mouse, bag, quality, keyboard, bit, time, computer, screen, case, battery, fit, sleeve, size, speakers, sound, buy, one
DPLR	laptop, price, mouse, product, bag, quality, screen, keyboard, time, computer, size, bit, case, battery, sleeve, drive, fit, buy, weight, design
DP-HITS	backpack, netbook, zipper, compartment, laptop, bag, computer, stand, product, padding, case, mouse, machine, design, screen, battery, lapdesk, size, fit, thing
DPLR-R	price, mouse, product, quality, bag, laptop, size, buy, time, keyboard, job, battery, machine, fit, computer, design, drive, case, zipper, screen

The same as Table 2 shows, in Table 3 "bit" and "one" are tagged as nouns by OpenNLP. DPF has more features such as "power", "screen", "pad" compared to DPLR. We can see in the results of LRCW, DPLR and DPLR-R methods, "price","mouse", "bag" are the most three important product features for laptop computers. However, candidate feature "laptop" is not a product feature.

In Table 4, DPLR-R gives "location", "room", "staff", "breakfast", and "place" as the top 5 most important service aspects. The candidate "walk" is not a service aspect. Although consumers like to use "minute walk" to describe the walking distance to somewhere from their hotels. DP-HITS has the similar results as DPLR-R, and the top 5 features are "breakfast", "bathroom", "bed", "stay", "shower".

Table 4. Top 20 Features of the Hotel Dataset

Approaches	Top 20 Features
DPF	hotel, room, staff, night, breakfast, location, place, bathroom, bed, time, day, floor, area, shower, reception, street, stay, service, station, city, price
FORank	hotel, room, rooms, staff, breakfast, bit, people, place, area, hotels, location, thing, bathroom, night, bed, service, stay, person, way, time, experience
LRCW	room, hotel, staff, location, rooms, breakfast, bathroom, bed, walk, place, area, bit, stay, street, value, service, beds, restaurants, night, price
DPLR	room, hotel, staff, location, breakfast, bathroom, bed, walk, place, area, street, bit, stay, value, service, price, shower, night, food, station
DP-HITS	hotel, breakfast, bathroom, bed, stay, shower, street, room, food, coffee, neighborhood, desk, kitchen, buffet, place, cafe, bedroom, sleep, terrace, area
DPLR-R	hotel, location, room, walk, staff, breakfast, place, value, bathroom, area, stay, street, city, bed, water, access, size, experience, service, shower

DPLR demonstrates five features including "room", "staff", "location", "breakfast", "bathroom" that attract most people's attentions, even more than "price". LRCW's result is similar to DPLR's result. "Restaurants" is not a service feature, but many consumers talk about restaurants around their hotels in their reviews. "Street" is not related to the hotel service too, however, consumers care about street noises. We can see in the results of LRCW and DPLR methods "room", "staff" are the two most important service features for hotel.

Although we have gotten different product features for different product domains, we can see there are some features are general, such as "price". In Tables 3 and 4, the feature "price" is in the top 20 features lists of laptop and hotel. "Price" seems to be the most general product feature.

Fig.1 shows the MAP distributions of the top 200

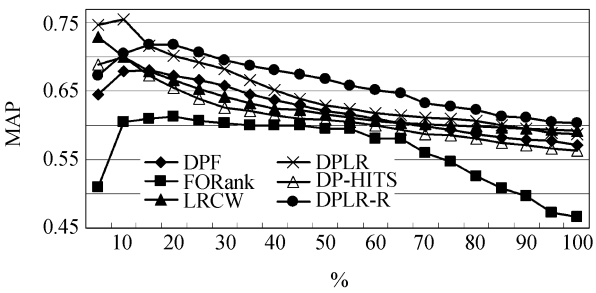

Fig.1. MAP distributions of feature ranking results on the phone dataset.

ranked candidate features of the phone dataset. Fig.2 gives the MAP distributions of the top 200 ranked candidate features of the laptop dataset. Fig.3 demonstrates the MAP distributions of the top 200 ranked candidate features of the hotel dataset. The x-axis in Figs. 1∼3 presents the percent of the top 200 candidate product features.

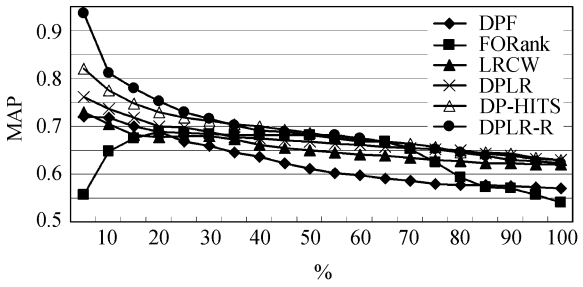

Fig.2. MAP distributions of feature ranking results on the laptop dataset.

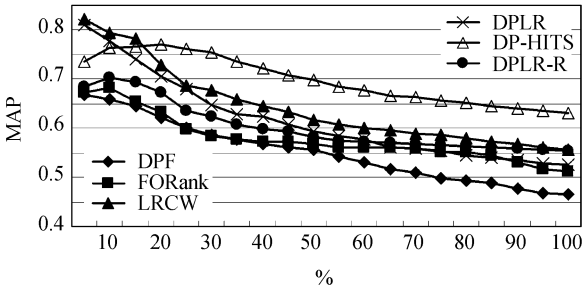

Fig.3. MAP distributions of feature ranking results on the hotel dataset.

In Fig.1, DPLR-R and DPLR have the best performances. DPLR has the better performance on the top 15% ranked candidate product features. DPLR-R and DPLR have better performance than DP-HITS. The performance of DPF approximates to the performance of LRCW. FORank has lower performance than other methods.

In Fig.2, DPLR-R has the best performance. The ranking result of DP-HITS is similar to the result of DPLR-R when the ranking percent is greater than 35%. FORank has good performance when the percent of the top 200 candidate features is between 30% and 45%. The result of LRCW and the result of DPLR are quite similar. DPF has the lowest MAP distribution.

In Fig.3, we can see DP-HITS has the best performance among all the proposed ranking methods. LRCW and DPLR have better ranking results than DPLR-R. DPF has the lowest performance. DPLR-R model performs worse than the DP-HITS model and DPLR model on the hotel dataset, and it also performs worse than the DPLR and LRCW models. The model cannot produce consistent results on our datasets.

In the hotel dataset, we find consumers are likely to tell stories about their traveling and accommodation experiences. In this case, consumers even do not mention service aspects directly. However, in the phone and the laptop datasets, consumers are likely to comment product features directly. Hence, we believe our simple product feature extraction rules are limited and the extracted results are more propitious to rank product features that are mentioned explicitly, for example, "good screen", "great battery". This is the reason that our model cannot produce consistent results on the hotel dataset.

Tables 5∼7 show ranking results with different initial weight vectors in our product feature ranking algorithm GetFeatureWeights. Here $\boldsymbol{w}$:0.01 means each element of the weight vector is initialized to 0.01. We can see

Table 5. Ranking Precisions (MAP) for Different Initial Weights on the Phone Dataset

Percent	$\boldsymbol{w}$:0.001	$\boldsymbol{w}$:0.005	$\boldsymbol{w}$:0.01	$\boldsymbol{w}$:0.1	$\boldsymbol{w}$:0.2	$\boldsymbol{w}$:0.5
10	0.7049	0.7204	0.7298	0.7049	0.5002	0.4209
20	0.7179	0.7255	0.7320	0.7179	0.4978	0.3892
30	0.6958	0.7003	0.7050	0.6958	0.4549	0.3132
40	0.6818	0.6844	0.6889	0.6818	0.4357	0.2752
50	0.6684	0.6724	0.6760	0.6684	0.4179	0.2437
60	0.6525	0.6633	0.6665	0.6525	0.4058	0.2354
70	0.6331	0.6543	0.6572	0.6331	0.3991	0.2199
80	0.6236	0.6409	0.6433	0.6236	0.3931	0.2154
90	0.6111	0.6277	0.6301	0.6111	0.3850	0.2124
100	0.6032	0.6180	0.6201	0.6032	0.3826	0.2107

Table 6. Ranking Precisions (MAP) for Different Initial Weights on the Laptop Dataset

Percent	$\boldsymbol{w}$:0.001	$\boldsymbol{w}$:0.005	$\boldsymbol{w}$:0.01	$\boldsymbol{w}$:0.1	$\boldsymbol{w}$:0.2	$\boldsymbol{w}$:0.5
10	0.8604	0.8570	0.8594	0.8115	0.7981	0.2585
20	0.7770	0.7765	0.7771	0.7531	0.7280	0.2453
30	0.7428	0.7422	0.7429	0.7167	0.6913	0.2377
40	0.7299	0.7305	0.7295	0.6919	0.6600	0.2421
50	0.7174	0.7153	0.7153	0.6829	0.6317	0.2489
60	0.7027	0.7021	0.7018	0.6755	0.6190	0.2551
70	0.6911	0.6906	0.6907	0.6548	0.5948	0.2603
80	0.6774	0.6770	0.6769	0.6464	0.5768	0.2627
90	0.6693	0.6689	0.6688	0.6297	0.5632	0.2644
100	0.6561	0.6557	0.6555	0.6232	0.5484	0.2661

Table 7. Ranking Precisions (MAP) for Different Initial Weights on the Hotel Dataset

Percent	$\boldsymbol{w}$:0.001	$\boldsymbol{w}$:0.005	$\boldsymbol{w}$:0.01	$\boldsymbol{w}$:0.1	$\boldsymbol{w}$:0.2	$\boldsymbol{w}$:0.5
10	0.7151	0.7151	0.7151	0.7022	0.6968	0.6662
20	0.6825	0.6813	0.6813	0.6731	0.6622	0.5072
30	0.6239	0.6232	0.6237	0.6229	0.6179	0.4770
40	0.6053	0.6047	0.6052	0.5988	0.5927	0.4585
50	0.5879	0.5874	0.5877	0.5819	0.5796	0.4437
60	0.5751	0.5747	0.5750	0.5735	0.5686	0.4280
70	0.5677	0.5674	0.5681	0.5671	0.5630	0.4184
80	0.5630	0.5627	0.5635	0.5621	0.5575	0.4067
90	0.5587	0.5584	0.5584	0.5579	0.5535	0.3981
100	0.5535	0.5533	0.5533	0.5548	0.5491	0.3880

the values of the initial weight vector will influence the ranking results. When the value of each element of the initial weight vector is initialized to 0.01, the ranking results look better than other cases.

4 Two Applications Based on Product Feature Ranking Approach

In this section, we give two applications based on product feature ranking. In the first application, we uses DPLR and DPLR-R to conduct overall rating decomposition. In the second application, we use both DPLR and DPLR-R to generate two kinds of consumer surveys. We want to show that the research of product feature ranking has its empirical value.

4.1 Overall Rating Decomposition

We use DPLR method and DPLR-R method respectively to carry out overall rating decomposition. The overall rating decomposition algorithm has four major steps as follows.

• Extract product features using DP algorithm and our proposed two rules from all reviews in the same product domain. Extract opinion units using the two rules too.

• Rank product features using DPLR or DPLR-R approach.

• Compute average ratings for opinion units using (2).

• Compute rating for a product feature using opinion units associated with the product feature using (3).

• For every review, generate feature tuples including features, opinion units and feature ratings.

Fig.4 is an example Web interface of overall rating decomposition for two reviews using DPLR method. In this example, the overall rating for the first phone review is 1. In the first review, we get four candidate phone features including "phone", "piece", "garbage" and "lock". Unfortunately the review is short and has not mentioned many features. Hence the extracted candidate features are not real phone features. However, the extracted information also can reflect the opinions about the phone, such as "garbage", "lock". In the second review, we get top 6 candidate product features which are "phone", "quality", "memory", "style", and "functionality". In this case, "phone" and "technology" are not product features. We also get the ratings for the extracted product features. For instance, "quality" has rating value 3.98304 which quite approximates to the overall rating.

Fig.5 is an example Web interface of overall rating decomposition for one reviews using DPLR-R method. In this example, the overall rating for the phone review is 2. In the showed review, we get six candidate phone

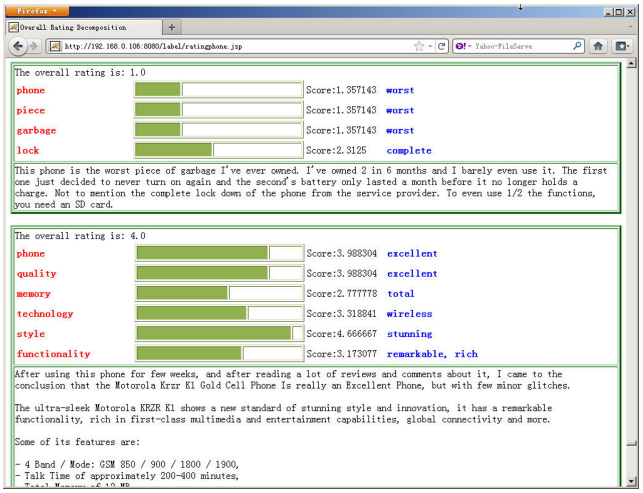

Fig.4. Example of overall rating decomposition using DPLR.

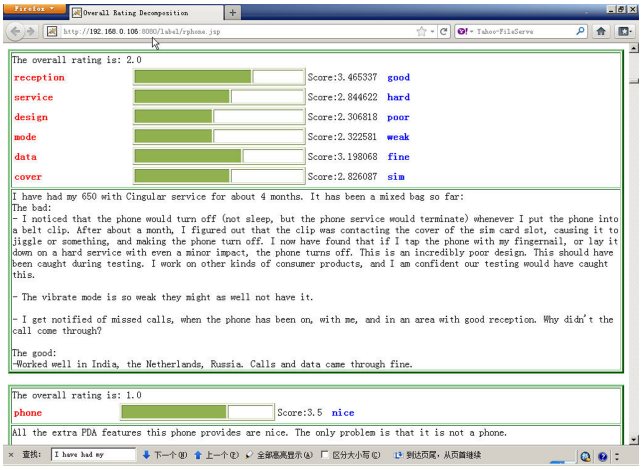

Fig.5. Example of overall rating decomposition using DPLR-R.

features including "reception", "service", "design", "mode", "data" and "cover". We also get the corresponding opinion units including "good", "hard", "poor", "weak", "fine" and "sim". The average ratings are ratings calculated for the opinion units.

4.2 Automatic Consumer Survey Generation

In traditional product and service marketing management, product and service providers design consumer surveys for getting consumers' feedbacks to evaluate product and service quality. SERVQUAL framework[13-14] is the most popular and ubiquitous service quality instrument[15]. Based on SERVQUAL, there are many product and service quality evaluation derivations for various investigation purposes. To design a proper and accurate consumer survey for a product or service we firstly need to know consumers are concerned with what product or service features. Traditional product or service surveys can be designed

by domain experts. However, surveys designed by humans may be subjective and it is hard to explain why and how a survey is constructed.

We try to generate consumer surveys by mining online reviews, since service consumers like to express opinions about products and services. Our survey generation process is automatic and simple. Based on our product feature ranking approach DPLR-R, after we acquire ranked candidate features we only generate product surveys using the top N ranked candidate product features. A survey entry is about a product feature, and the choices of a survey entry are opinion units that are extracted based on DP algorithm and they are in the context windows of the candidate product features. Every survey entry has five choices. Fig.6 is an example of automatically generated consumer survey from consumer reviews. Actually, we can modify some survey items to let the survey be more suitable, for example to remove incorrectly survey items (product features). But here we only show the generated survey. Firstly we extract all candidate product features and candidate opinion units using double propagation algorithm, then we use DPLR or DPLR-R to rank the candidate product features. We get the average overall rating for each candidate opinion unit using (2). We divided overall rating range (for example from 1 to 5) into five intervals, then we select representative opinion unit for these rating intervals of each product feature. We only select the candidate an opinion unit with the largest

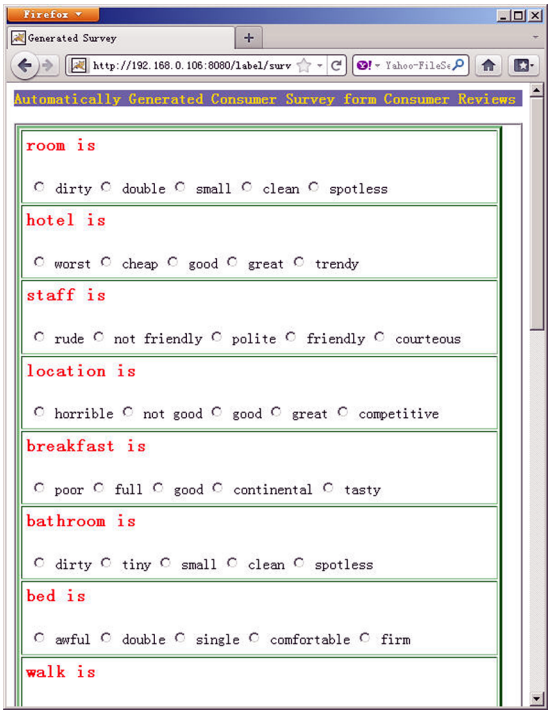

Fig.6. Example of automatically generated consumer survey using DPLR on the hotel dataset.

number in its corresponding rating interval as the representative opinion unit. The selected opinion units are listed as choices of one survey item. Our previous work[3] has given a method to generate service survey automatically. Previous work[3] only considers adjectives and nouns extracted using simple extraction rules, while in our present work, consumer surveys are generated based on DPLR method or DPLR-R method.

In a traditional customer survey, there may be a couple of questions for consumers to answer. The common way of doing so is to provide several candidate answer choices for consumers to select. For instance a typical survey item of phone screen is

Screen is:
A: Very poor B: Poor C: Neural D: Good E: Excellent.

After we rank the product features, it is easy to generate a survey containing survey items similar to the above one. In this case, answer choices (Poor, Good, etc.) of this kind of survey are the same for different candidate product features.

There is another alternative form for answer choices of a survey item. A typical example of this kind of survey item is

Bathroom is:
A: Filthy B: Dirty C: Small D: Clean E: Spotless,

where we use opinion units to rate the product feature or service aspect. In this case we call answer choices representative opinion units. For example, in above example, answer choice "filthy" is a representative opinion unit.

Suppose that ratings of opinion units range from 1 to t, where t is a real number. In this work, because ratings of reviews range from 1 to 5, ratings of opinion units range from 1 to 5 too. That is to say $t = 5$. We can select m representative opinion units for a candidate product feature, here, $m = 5$. In other words, we must select m representative opinion units to be answer choices for a candidate product feature. It is obvious, a candidate product feature may have more than m kinds of opinion units according to rating ranges of opinion units. If $1 \leqslant j < m$, the j-th opinion unit's rating range is $[1 + (j-1) \times (t-1)/m,\ 1 + j \times (t-1)/m)$ (it means $1 + (j-1) \times (t-1)/m \leqslant r < 1 + j \times (t-1)/m$, where r is rating for the opinion unit), and if $j = m$, the rating range is $[1+(j-1)\times(t-1)/m,\ 1+j\times(t-1)/m]$ (it means $1 + (j-1) \times (t-1)/m \leqslant r \leqslant 1 + j \times (t-1)/m$, where r is rating for the opinion unit), where $1 \leqslant j \leqslant m$. We use major opinion unit selection method to select representative opinion units. The strategy relies on co-occurrences of candidate product features and their related candidate opinion units. It tries to select the j-th

opinion unit from the j-th rating range with the largest co-occurrences with the candidate product feature f_i in all sentence.

In Fig.6, we can see the most six important service aspects including "room", "staff", "location", "breakfast", "bathroom", and "bed". However, the survey is not perfect. For instance, "hotel" is not a service aspect, and it should not be a survey entry. Service aspect "room" almost has the same opinion choices as service aspect "bedroom", while people really care about the sanitary conditions of rooms and bathrooms. The "staff" survey entry's second choice is "not friendly", because we use opinion unit to be survey choices that may include negative indicators. The "location" survey entry has a choice "horrible" which is opposite to the choice "competitive". For the "bed" survey entry, "double" or "single" should not be opinion units. If two survey entries have the same stem, for example "rooms" and "room", they should be integrated as the same survey entry. Duplicated survey entries about the same feature can be consolidated and integrated through combining opinion units.

Fig.7 is a survey example generated using DPLR-R method. The service aspects are ranked according to DPLR-R. Every service aspect has five choices. The survey is generated automatically, hence in Fig.7, some candidates are not service aspects such as "hotel" and "walk". In Fig.7, candidate answer choices of a candidate product feature are the same for another candidate product feature.

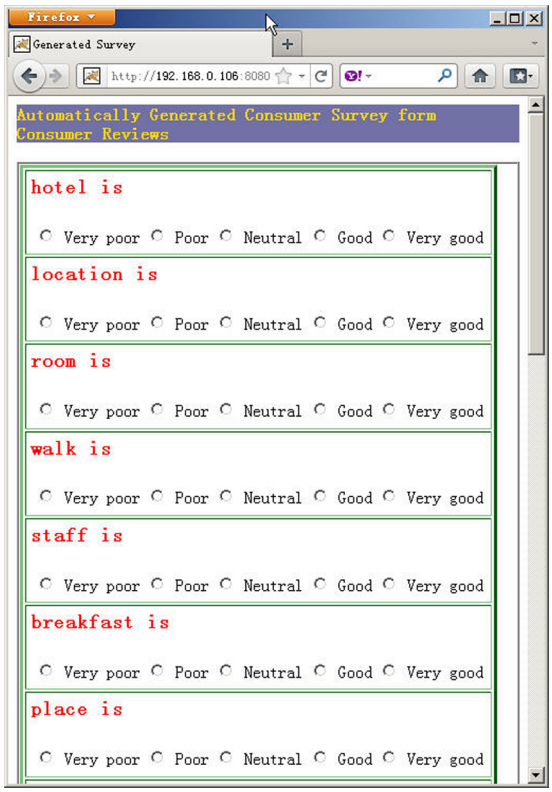

Fig.7. Example of automatically generated consumer survey using DPLR-R on the hotel dataset with fixed item choices.

5 Related Work

Recent years there is a considerable amount of research on feature-based product opinion mining.

Hu and Liu[11] and Liu *et al.*[16] are early research effort that summarized feature-based product opinions through applying association mining. The work[11] assumes that consumers likely use similar words to comment the same product. Hence, words such as nouns with high frequency have high probability to be product features. [11] employed association mining algorithm[17] to get candidate product features. The work[18] also mined product reputations on the Web and gave an approach of automatically generating words or phrases to describe product reputations. Popescu and Etzioni[19] proposed an unsupervised information-extraction system Opine which found important product features. In order to find explicit product features, Su *et al.*[20] proposed a mutual reinforcement approach to cluster product features and opinion words simultaneously and iteratively by fusing both their content information and sentiment link information. Gamon *et al.*[21] used TF-IDF (term frequency-inverse document frequency) related approach to mine topics and sentiment orientation jointly from free text consumer feedback. Pang's research survey[22] gave overall promising techniques for implementing empirical opinion mining systems.

There are several research publications focusing on product feature ranking problem. Lu *et al.*[23] studied the problem of generating a "rated aspect summary" of short comments coming with an overall rating. Lu *et al.*[23] gave a PLSA-based method[24] to conduct finding major product aspects and overall rating decomposition through aspect clustering. Li *et al.*[25] uses information gain method to ranking service aspects without clustering aspect. Wang *et al.*[26] addressed the Latent Aspect Rating Analysis text mining problem and they proposed a probabilistic rating regression model to discover each individual reviewer's latent opinion on each aspect. Zhang *et al.*[4] extracted product feature using double propagation algorithm as our work, but they adopted HITS[12] to rank the product features. Some research focuses on sentiment summarization.

6 Discussion

If we can get good sentiment summarization about a product, then it is possible to rank the product features based on the summarization. Some research publications are mainly about topic mining from reviews.

Yi *et al.*[27] extracted opinion about a subject from online text documents using mixture language model and likelihood ratio. Mei *et al.*[28] tried to model facets and opinions topic mixture in weblogs. Titov and McDonald's work[29] is another excellent topic-oriented aspect extraction publication. However, our work tries to rank product features, and we are not going to cluster them into topics. We believe topic mining techniques will be helpful for our future work.

Sentiment classification has attracted many research efforts. Our work has strong relation with word sentiment classification. Hatzivassiloglou[30] predicted adjective's sentiment orientation by using conjunction constrains on conjoined adjectives. Turney[31] used PMI method to judge the sentiment orientation of words. We think word sentiment classification research can benefit our work and may improve the feature ranking performance. Lerman *et al.*[32] built a sentiment summarizer by training a ranking SVM (support vector machine) model over the set of human preference judgments. The sentiment summarizer can be used to rank product features.

Text classification and clustering may play an important role in finding good, important, and helpful consumer reviews. It is possible for us to conduct sentiment classification if we can find some explicit and implicit links among consumer reviews. Our consumer reviews are extracted from Web pages, hence hyperlinks are explicit links. Furthermore, implicit links can be constructed by similarity or click-through paths among these consumer reviews. Shen *et al.*[33] showed implicit links can improve performance of Web page classification. Web users have social relations when they publish consumer reviews, therefore, the social relations among Web users may be used to find important or helpful consumer reviews too. Ma *et al.*[34] proposed a probabilistic factor analysis framework to recommend with explicit and implicit social relations. We believe these excellent approaches can be used in the sentiment classification. Furthermore, implicit links among text can be used for extracting keyphrases from documents. Wan and Xiao[35] proposed a graph-based ranking algorithm which makes use of both the local information in the specified document and the global information in the neighbor documents. This is another way for us to rank product features, if the features are in the list of keyphrases extracted from consumer reviews. We think it may be a good way to combine social links and text links together to rank product features. In the future, we will seek a novel graph-based product feature ranking method with which graphs are constructed based on extracted explicit and implicit links among reviews and product features.

We believe clustering of the opinion words and the features of products is an important research aspect in the field of sentiment analysis and opinion mining. In our method we only adopt simple heuristic rules to extract candidate product features, service aspects and candidate opinion words. We have not considered methods of clustering different product features. Some excellent research focuses on product feature clustering, for example Zhai *et al.*'s work[36], Guo *et al.*'s work[37]. Guo *et al.*[37] proposed a two-stage method to implement product feature categorization with multilevel latent semantic association. In [37], at the first stage, they firstly constructed the first latent semantic association (LaSA) model to group words into a set of concepts; at the second stage, they categorized the product features. We think these proposed product feature clustering approaches can be used in our work to improve product feature ranking accuracy.

There are some possible ways to rank product features. We can use machine learning methods to train ranking models to rank product features. The challenge lies in that we need to extract some useful training features for these ranking models.

7 Conclusions

This paper focuses on product feature ranking. We conduct linear regression on product features using ratings of opinion units and overall ratings. We employ a gradient decent algorithm to get a local optimal solution for the regression. Our approach (DPLR-R) is a two-stage method. At the first stage, we extract product features using the state-of-the-art product feature extraction algorithm double propagation algorithm and two rules together. We also use our two rules to extract opinion units. At the second stage, we regress on the extracted product features and opinion units by exploiting overall ratings of reviews. To conduct comparisons, we compare DPLR-R with Double Propagation-Based Linear Regression (DPLR), Linear Regression with Context Windows (LRCW) and PageRank-Based FORank method. Furthermore, we compare our method with the state-of-the-art method DP-HITS[4]. Experiments show DPLR-R is effective, and DPLR-R can rival the state-of-the-art approach DP-HITS[4]. It outperforms DPF, FORank, LRCW, and DPLR on our phone and laptop datasets, but DP-HITS has the best performance on our hotel dataset. DPLR-R can even outperform the state-of-the-art method DP-HITS on some datasets. Finally, we demonstrate two applications based on our two product feature ranking schemes DPLR and DPLR-R.

We use a linear regression-based method to rank

product features, however, there may be more sophisticated methods to rank product features, such as ranking SVM model or other supervised machine learning based models. And it is possible to extract and rank product features by utilizing sentiment summarizations information.

We must emphasize that our approach is a straightforward approach. There are some limitations using our approach. For example, the approach needs ratings of consumer reviews. But many consumer reviews have no corresponding ratings. Of course, because whether a product feature candidate is a true product feature is classified by humans, the precisions of ranked product features seem to be subjective. However, the evaluation conditions (the same judges and the same criterion) conducted on all the datasets are the same. Another interesting problem is whether we can apply our linear regression methods on the results of other product feature extract methods.

In the future, we will continue our research and we expect to find more effective and promising product feature ranking methods.

Acknowledgment We thank the anonymous reviewers for their constructive comments.

References

[1] Li S, Guan Z, Tang L, Chen Z. Exploiting consumer reviews for product feature ranking. In *Proc. the 3rd Workshop on Social Web Search and Mining* (*SWSM2011*), Beijing, China, July 28, 2011, Article No.13.

[2] Li S, Hao J, Chen Z. Graph-based service quality evaluation through mining Web reviews. In *Proc. the 2010 NLP-KE*, Beijing, China, Aug. 21-23, 2010, pp.280-287.

[3] Li S, Chen Z. Exploiting web reviews for generating customer service surveys. In *Proc. the 2nd International Workshop on Search and Mining User-Generated Contents*, Toronto, Canada, Oct. 26-30, 2010, pp.53-62.

[4] Zhang L, Liu B, Lim S H, O'Brien-Strain E. Extracting and ranking product features in opinion documents. In *Proc. the 23rd COLING*, Beijing, China, Aug. 23-27, 2010, pp.1462-1470.

[5] Qiu G, Liu B, Bu J, Chen C. Expanding domain sentiment lexicon through double propagation. In *Proc. the 21st International Joint Conference on Artificial Intelligence*, San Francisco, USA, July 11-17, 2009, pp.1199-1204.

[6] Liu B. Web Data Mining: Exploring Hyper-Links, Contents, and Usage Data. Springer Press, 2006, pp.411-449.

[7] Wilson T, Wiebe J, Hoffmann P. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proc. HLT 2005*, Vancouver, Canada, Oct. 6-8, 2005, pp.347-354.

[8] Hastie T, Tibshirani R, Friedman J H. The Elements of Statistical Learning. New York: Springer New York, 2001.

[9] Brin S, Page L. The anatomy of a large-scale hypertextual Web search engine. *Comput. Netw. ISDN Syst.*, 1998, 30(1-7): 107-117.

[10] Golub G H, van V L C F. Matrix Computations. The Johns Hopkins University Press, 1983.

[11] Hu M, Liu B. Mining and summarizing customer reviews. In *Proc. the 10th KDD*, Seattle, USA, Aug. 22-25, 2004, pp.168-177.

[12] Kleinberg J M. Authoritative sources in a hyperlinked environment. *J. ACM*, 1999, 46(5): 604-632.

[13] Parasuraman A, Zeithaml V A, Berry L L. SERVQUAL: A multiple-item scale for measuring consumer perceptions. *Journal of Retailing*, 1988, 64(1): 12-40.

[14] Parasuraman A, Zeithaml V A, Berry L L. A conceptual model of service quality and its implications for future research. *Journal of Marketing*, 1985, 49(4): 41-50.

[15] Landrum H, Prybutok V, Zhang X, Peak D. Measuring IS system service quality with SERVQUAL: Users' perceptions of relative importance of the five SERVPERF dimensions. *Informing Science: the International Journal of an Emerging Transdiscipline*, 2009, 12:17-35.

[16] Liu B, Hu M, Cheng J. Opinion observer: Analyzing and comparing opinions on the Web. In *Proc. the 15th WWW*, Chiba, Japan, May 10-14, 2005, pp.342-351.

[17] Agrawal R, Imielinski T, Swami A. Mining association rules between sets of items in large databases. In *Proc. SIGMOD 1993*, Washington, USA, May 25-28, 1993, pp.207-216.

[18] Morinaga S, Yamanishi K, Tateishi K, Fukushima T. Mining product reputations on the Web. In *Proc. KDD 2002*, Edmonton, Canada, July 23-25, 2002, pp.341-349.

[19] Popescu A, Etzioni O. Extracting product features and opinions from reviews. In *Proc. HLT/EMNLP*, Vancouver, Canada, Oct. 6-8, 2005, pp.339-346.

[20] Su Q, Xu X, Guo H, Guo Z, Wu X, Zhang X, Swen B, Su Z. Hidden sentiment association in Chinese web opinion mining. In *Proc. WWW 2008*, Beijing, China, April 21-25, 2008, pp.959-968.

[21] Gamon M, Aue A, Corston-Oliver S, Ringger E K. Pulse: Mining customer opinions from free text. In *Proc. IDA 2005*, Madrid, Spain, Sept. 8-10, 2005, pp.121-132.

[22] Pang B, Lee L. Opinion mining and sentiment analysis. *Found. Trends Inf. Retr.*, 2008, 2(1-2): 1-135.

[23] Lu Y, Zhai C, Sundaresan N. Rated aspect summarization of short comments. In *Proc. WWW 2009*, Madrid, Spain, April 20-24, 2009, pp.131-140.

[24] Hofmann T. Unsupervised learning by probabilistic latent semantic analysis. *Mach. Learn.*, 2001, 42(1-2): 177-196.

[25] Li S, Hao J, Chen Z. Generating tags for service reviews. In *Proc. ADMA 2010*, Chongqing, China, Nov. 19-21, 2010, pp.463-474.

[26] Wang H, Lu Y, Zhai C. Latent aspect rating analysis on review text data: A rating regression approach. In *Proc. KDD 2010*, Washington, DC, USA, July 25-28, 2010, pp.783-792.

[27] Yi J, Nasukawa T, Bunescu R, Niblack W. Sentiment analyzer: Extracting sentiments about a given topic using natural language processing techniques. In *Proc. ICDM 2003*, Nov. 19-22, 2003, pp.427-434.

[28] Mei Q, Ling X, Wondra M, Su H, Zhai C. Topic sentiment mixture: Modeling facets and opinions in weblogs. In *Proc. WWW 2007*, Banff, Canada, May 8-12, 2007, pp.171-180.

[29] Titov I, McDonald R. Modeling online reviews with multi-grain topic models. In *Proc. WWW 2008*, Beijing, China, April 21-25, 2008, pp.111-120.

[30] Hatzivassiloglou V, McKeown K R. Predicting the semantic orientation of adjectives. In *Proc. the 18th EACL*, Madrid, Spain, July 7-12, 1997, pp.174-181.

[31] Turney P, Littman M L. Measuring praise and criticism: Inference of semantic orientation from association. *ACM Trans. Inf. Syst.*, 2003, 21(4): pp.315-346.

[32] Lerman K, Blair-Goldensohn S, McDonald R. Sentiment summarization: Evaluating and learning user preferences. In *Proc. the 12th EACL*, Athens, Greece, March 30-April 3, 2009, pp.514-522.

[33] Shen D, Sun J, Yang Q, Chen Z. A comparison of implicit and explicit links for web page classification. In *Proc. WWW 2006*, Edinburgh, Scotland, UK, May 23-26, 2006, pp.643-650.

[34] Ma H, King I, Lyu M R. Learning to recommend with explicit and implicit social relations. *ACM Trans. Intell. Syst. Technol.*, 2011, 2(3): Article No.29.

[35] Wan X, Xiao J. Exploiting neighborhood knowledge for single document summarization and keyphrase extraction. *ACM Trans. Inf. Syst.*, 2010, 28(2): Article No.8.

[36] Zhai Z, Liu B, Xu H, Jia P. Clustering product features for opinion mining. In *Proc. the WSDM 2011*, Hong Kong, China, Feb. 9-12, 2011, pp.347-354.

[37] Guo H, Zhu H, Guo Z, Zhang X, Su Z. Product feature categorization with multilevel latent semantic association. In *Proc. CIKM 2009*, Hong Kong, China, Nov. 2-6, 2009, pp.1087-1096.

Su-Ke Li is currently an assistant professor in Peking University. He received his B.S. degree from North China Electric Power University. He received the M.S. and Ph.D. degrees from Peking University, in 2002 and 2012 respectively. His research interests include financial data mining, embedded system, Web mining and retrieval, opinion mining, social network, and information security.

Zhi Guan received his B.S. degree from Dalian University of Technology and Ph.D. degree from the Peking University, in 2002 and 2009 respectively. He is currently an assistant professor at Peking University. His research interests include applied cryptography, security and privacy of RFID.

Li-Yong Tang is an associate professor of Peking University. He received the B.S. and M.S. degrees in computer science from Nanjing University of Aeronautics and Astronautics, Nanjing, Jiangsu, China, in 1993 and 1996, respectively, and the Ph.D. degree in computer science from Peking University, Beijing, China, in 2000. His research interests include network and information security, system software, peer-to-peer network, social network, wireless networks and mobile applications.

Zhong Chen is currently a professor of School of Electronics Engineering and Computer Science, and director of the Network and Information Security Research Group of the Institute of Software, Peking University. He received his B.S. and Ph.D. degrees from Peking University, in 1983 and 1988 respectively. He has wide interests in network and information security, system software and embedded system, software/hardware co-design methodology and domain-specific software engineering.