Extracting Chinese Product Features: Representing a Sequence by a Set of Skip-Bigrams

Ge Xu[1,2,3], Chu-Ren Huang[1], and Houfeng Wang[2]

[1] Faculty of Humanities, The Hong Kong Polytechnic University, Hong Kong
churenhuang@gmail.com
[2] Institute of Computational Linguistics, Peking University, Beijing, 100871
wanghf@pku.edu.cn
[3] Department of Computer Science, MinJiang University, Fuzhou, 350108
xuge@pku.edu.cn

Abstract. A skip-bigram is a bigram that allows skips between words. In this paper, we use a set of skip bigrams (a SBGSet) to represent a short word sequence, which is the typical form of a product feature. The advantage of SBGSet representation for word sequences is that we can convert between a sequence and a set. Under the SBGSet representation we can employ association rule mining to find frequent itemsets from which frequent product features can be extracted.For infrequent product features, we use a pattern-based method to extract them. A pattern is also represented by a SBGSet, and contains a variable that can be instantiated to a product feature.We use two data sets to evaluate our method. The experimental result shows that our method is suitable for extracting Chinese product features, and the pattern-based method to extract infrequent product features is effective.

Keywords: sentiment analysis, product feature, word sequence, skip-bigram.

1 Introduction

In sentiment analysis, an important task is to extract people's opinions expressed on features of a product. Such information is valuable for both consumers and product manufacturers. For this task, the first step is to provide a set of product features. In this paper, we mainly focus on extracting product features from online Chinese product reviews.

In [1,2], the authors use association rule mining to obtain frequent itemsets, which are then pruned to find product features. However, association rule mining is unable to consider the order of words, which is very important in natural language texts. In our paper, we want to keep the order of words when using association rule mining, thus can obtain sequences of words of variable lengths, which are the candidates of product features.

For extracting Chinese product features, some problems can affect the performance.

First, although word segmentation and POS tagging in Chinese can reach over 90% accuracy, for domain-specified terms, the performance is less reliable. In Table 1, we offer some terms of digital cameras suffering from wrong word segmentation and POS

Table 1. Examples of wrong word segmentation and POS tagging on digital cameras

w_1	w_2	meaning
热/a	靴/ng	hot shoe
微/ag	距/vg	macro
微/dg	距/v	macro
对/p	焦/j	focusing
滤/v	镜/ng	filter
像/v	素/dg	pixel
防/v	抖/v	anti-shake

tagging. Especially, it is noted that 微距 (macro) has two different results of POS tagging, which shows the difficulty to identify unknown domain-specific terms.

Second, in current Chinese product feature extraction, *noun*+ (a noun or a sequence of nouns) is normally used to extract product feature candidates. Although *noun*+ facilitates the following processing, it excludes many product features that include verbs or other POSs (mainly due to error POS tagging).

These two problems are related. Since we do not have satisfying word segmentation and POS tagging for Chinese domain-specific terms, we hesitate to use Chinese parsers that are based on word segmentation and POS tagging. So we choose to extract product features by *noun*+ which require no parsing analysis.

To handle the above problems, we propose to use a set of skip bigrams (SBGSet) to represent a word sequence. A skip bigram (SBG) is a bigram that allow skips between the two words in the bigram. A word sequence and its corresponding SBGSet can be converted from each other. When combined with Apriori algorithm, it is convenient to extract frequent product features. Under this framework, wrong segmentation is corrected in the same way as multi-word phrases are identified, and filtering rules can be designed flexibly. Furthermore, we use a pattern-based method to extract infrequent product features, and a pattern is also represented as a SBGSet.

We will the give detailed introduction to our approach in the following sections.

2 Related Work

In [1,2], from a large number of reviews, the authors at first identify nouns and noun phrases using a parser, then their frequencies are counted and only the frequent ones are kept since important product features are more likely discussed by consumers. Furthermore, two pruning methods are used to improve the precision and recall. In [3], the authors use labeled data to mine Class Sequential Rules (CSR), which are then converted to language patterns for extracting product features.

Some attempts to improve the recall of product features are performed. In the paper of [4], the authors use nouns, noun phrases and verb phrases to extract feature candidates. The adding of verb phrases helps to identify more product features, and also incurs more noises. In [5], the authors claim that using noun phrases as product feature candidates limits the recall of product feature extraction, and propose a novel approach by generalizing syntactic structures of the product features.

In [6], the authors try to remove those noun phrases that do not have meronymy (part of) relationship with the target product. For example, the meronymy discriminators for the scanner class are, “of scanner”, “scanner has”, “scanner comes with”, etc., which are used to find components or parts of scanners by searching on the Web. The algorithm also distinguishes parts from properties using WordNet’s *is-a* hierarchy and morphological cues.

The double propagation method described in [7] can also be used to extract product features, which was highly dependent on a syntactic parser. The method started with only a set of seed opinion words (no seed features are required) and utilized the association or dependency relations between opinion words and features. The mutual reinforcement between feature candidates and opinion words was also used in a modified HITS algorithm [8].

In recent years, there has been an increasing interest in Chinese product feature extraction. The method in [9] starts from a small seed set of opinion words, and identifies product features by opinion words and vice verse through a bootstrapping iterative learning strategy; In [10], the authors propose a mutual reinforcement method to cluster both product feature candidates and opinion words. An association matrix between product features and opinion words was constructed. Together with traditional similarity approaches, the association matrix was used to calculate the similarity between homogeneous nodes. A similar iterative method also occurred in [11], which clustered product features and opinion words simultaneously by fusing both semantic information and co-occurrence information.

3 Representing a Word Sequence with a Set of Skip Bigrams

3.1 A Set of Skip Bigrams

Skip-grams are n-grams that allow words to be “skipped”. In [12], the authors define k-skip-n-grams for a sentence $w_1, w_2, ...w_m$ to be the set $\{w_{i_1}, w_{i_2}, ...w_{i_n} | \sum_{j=1}^{n} i_j - i_{j-1} <= k + 1 \; and \; i_j - i_{j-1} > 0\}$. For example, ”4-skip-2-gram” results generalized bi-grams include 4 skips, 3 skips, 2 skips, 1 skip, and 0 skips (typical bi-grams formed from adjacent words).

In our paper, we only consider **skip-bigram** (SBG) because it is possible that we combine a set of skip-bigrams (SBGSet) to a sequence of any length. A SBG is denoted as $(word_1 \; word_2)$.

For example, for the following sequence of words,

“The price is high”,

the SBGSet is {(The price), (The is), (The high), (price is), (price high), (is high)}. Furthermore, in most cases, we can recover the sequence using SBGSet. For example, given the SBGSet {(The price), (The is), (The high), (price is), (price high), (is high)}, “The price is high” can be recovered uniquely.

In addition, it is possible that the words in a sequence recovered from a SBGSet are not adjacent. For example, the SBGSet {(The price), (The high), (price high)} can be converted to “The price high”, in which “price” and “high” are not adjacent in original sentence “ The price is high”.

3.2 The Conversion between a Sequence and a SBGSet

From a Sequence to a SBGSet. It is straight forward to convert a word sequence to a SBGSet. For our experiments on extracting Chinese product features, since product features are normally single words or short phrases, and also opinions occur with product features in local texts, we only consider skip-bigrams which has at most three skips between the two words.

For example, for the word sequence "abcdef" (a letter is seen as a word), the SBGSet generated is $\{$(a b), (a c), (a d), (a e), (b c), (b d), (b e), (b f), (c d), (c e), (c f), (d e), (d f), (e f)$\}$.

From a SBGSet to a Sequence. Definition 1: valid SBGSet
A valid SBGSet is the SBGSet that can be converted to a word sequence uniquely, and no abundant SBGs exist.

The necessary condition for a valid SBGSet is :

1. There exists an integer m, and the size of the SBGSet is $m * (m - 1)/2$.
2. The number of terms[1] in the SBGSet is less than or equal to m. If there exists a word that occurs more than once in the original sequence, the number of terms is less than m, otherwise equal to m.

The above condition can not guarantee that a SBGSet is a word sequence. For example, $\{$(a b), (b c), (c a)$\}$ meets the necessary condition, however, no word sequence can be recovered.

The necessary condition for a valid SBGSet is used to eliminate invalid SBGSets, which is not supposed to recover a word sequence. For a SBGSet that meets the necessary conditions, we simply consider all the permutation of the terms in the SBGSet, and keep the one that satisfies all the order relationship formed by SBGs in the SBGSet. Since a product feature is a short sequence of words, such processing is affordable.

However, it is possible that a valid SBGSet can be converted to multiple word sequences. For this we give the following theorem.

Theorem 1: If at most one word in a sequence occur more than once, the SBGSet generated by the sequence can be uniquely converted to the sequence.

The proof is not shown due to space limit.

When two or more words in a sequence occur more than once, it is possible that the generated SBGSet from the sequence can be converted to more than one sequence. For example, the generated SBGSet for "baab" is $\{$(b a), (b a), (b b), (a a), (a b), (a b)$\}$. However, this SBGSet can be recovered to two sequences: "baab" and "abba".

When we see a product feature as a word sequence, normally the length of the product feature is short, and the possibility that two or more words occur more than once is extremely low. So, under this observation, we can always get the original sequence of words for a product feature by the corresponding SBGSet.

Another problem may affect the conversion from SBGSet to a word sequence. It is possible that the SBGs in a valid SBGSet are far separated. For example, assume that we have a SBGSet $\{$(a b), (b c), (a c)$\}$, which can be recovered to the word sequence "abc".

[1] If a word occur more than once, it is seen as one term.

However, the three SBGs (a b), (b c), and (a c) can be far apart, such as in a sentence like "XXXXabXXXXbcXXXXac", so the possibility that "abc" is a semantic concept is low. Although the probability exists, no such cases are observed in our experiments, so we choose to ignore this.

4 Extracting Frequent Product Features

4.1 Creating SBGSet Corpus

For our experiments, each line of text is a sentence after preprocessing, which is seen as a word sequence, and then convert to a SBGSet. To processing single words simultaneously, we create a special type of SBG for any single word in the form of $(w = w)$, where w is a single word, see Table 2 for details.

Table 2. Examples of converting word sequences to SBGSets

sequence	quliaty of the battery
SBGSet	(quliaty=quliaty), (of=of), (the=the), (battery=battery), (quliaty of), (quliaty the), (quliaty battery), (of the), (of battery), (the battery)
sequence	quliaty of that bad battery
SBGSet	(quliaty=quliaty), (of=of), (that=that), (bad=bad), (battery=battery), (quliaty of), (quliaty that), (quliaty bad), (quliaty battery), (of that), (of bad), (of battery), (that bad), (that battery), (bad battery)
sequence	quliaty of battery is
SBGSet	(quliaty=quliaty), (of=of), (battery=battery), (is=is), (quliaty of), (quliaty battery), (quliaty is),(of battery),(of is), (battery is)

After the conversion, we have a SBGSet corpus, in which each line (a sentence) is represented a set, not a sequence, and can be seen as a transaction for Apriori algorithm. It should be mentioned that in the SBGSet corpus, a SBGSet (a line) normally is not a valid SBGSet on the whole (because we restrict skips between words), but contain many subsets which are valid SBGSet.

4.2 Extracting Frequent Candidate Product Features

In the SBGSet corpus, we see a SBG as an item, and a SBGSet generated from a line of text as a transaction. Thus, we can use the association rule mining to extract frequent itemsets (SBGSets), and then recover short word sequences that are possible product features.

In association rule mining, $I = i_1, i_2, ..., i_n$ is a set of items, and D is a set of transactions. Each transaction contains a subset of I. An association rule is an implication of the form $X \rightarrow Y$, where $X \subset I, Y \subset I, X \cap Y = \emptyset$.

As in [1], we use Apriori algorithm in [13] to find all frequent itemsets in the transaction set.

For the example in Table 2, after running Apriori algorithm with minimum support 3, we obtain 41 itemsets (SBGSets). Of them, many are invalid SBGSet and can not be converted to word sequences. After converting valid SBGSets to sequences, we have sequences of words in Table 3.

Table 3. Recovered sequences of words

sequences of words	support
quality	3
quality of battery	3
quality of	3
quality battery	3
of	3
of battery	3
battery	3

4.3 Filtering Product Features

After getting word sequences, we use some filtering rules to reduce the size of possible product features. For example, from Table 3, sequences such as "of", "of battery" are not product features and should be eliminated. The filtering rules are highly linguistic-dependent, for our experiments, we use the following filtering rules:

- A product feature contains 2∼8 Chinese characters.
- A product feature contains 1∼3 Chinese words.
- Remove the sequences that contain characters other than Chinese characters.
- POS filtering: We define a set of error POS tags (applied on all words) and a set of valid POS tags (precisely nouns and verbs, applied on the words contain at least two Chinese characters). In addition, if the last word in a product feature contain more than one Chinese character, the word must be a noun.
- Word filtering: Some words are not supposed to appear in product features, normally they are stop words such as 是 (be), 的 (of), 这 (this), 有 (have) etc.

Since a feature may be a part of another feature, we use p-support to recalculate the frequency of a feature. In [1], the authors defined that **p-support** of a feature *ftr* is the number of sentences that *ftr* appears, and these sentences must contain no features that contain *ftr*.

5 Extracting Infrequent Product Features by Patterns

We can extract more itemsets (possible product features) by decreasing the support for Apriori algorithm. However, when the support is very low, the size of extracted itemsets is quite large and makes the processing unaffordable.

In this paper, to extract infrequent product features, we use the existing product features to find the patterns, and then use these patterns to extract new product features. Since a pattern is also a sequence, we can also represent a pattern using a SBGSet.

5.1 Selecting Seed Product Features

To obtain patterns for extracting product features, we need some high-quality product features as seed product features. These seeds are used to obtain the contexts in which product features occur, then we extract product features using these contexts.

There are many ways to obtain high-quality product features. For example, we can rank the frequent product feature, and choose the ones with high scores as product features. In this paper, since the focus is not ranking product feature and the size of frequent product feature is small (normally several hundreds after filtering), we simply manually label the frequent product features to get seed product features.

5.2 Creating the SBGSet Corpus for Extracting Patterns

Having seed product features, we use the following steps to obtain patterns:

1. Create the SBGSet corpus by extracting sentences which contain seed product features. Note that a seed product feature in a sentence is replaced by a variable X.
2. Use Apriori algorithm to extract frequent SBGSets.
3. SBGSets are filtered and converted to patterns

Definition 2: pattern
A *pattern* is a sequence with a *variable* and at least one word.

Definition 3: variable
A *variable* can be instantiated as a word sequence.

We provide an example to illustrate the steps.

Suppose we already have seed product features, and one of the product features is "touch screen". Table 4 shows how we replace "touch screen" in the line with a variable X, and get a SBGSet containing a variable X.

Table 4. Create SBGSet corpus for extracting patterns

The touch screen is big
$\Downarrow$
The X is big.
$\Downarrow$
{(The X), (The is), (The big), (X is), (X big), (is big)}

To get the SBGSet corpus for extracting patterns, we go thorough the whole text file, and extract all the lines that contain any seed product feature; all the product features will be replaced by the variable X.

We also create two pseudo-words (START and END) for the beginning and ending of a sentence respectively, which is useful in extracting patterns for infrequent product features.

5.3 Extract Frequent Patterns

Then, each line in SBGSet corpus created in last section is again represented by SBGSet, and can be seen as a transaction for Apriori algorithm, and we use the algorithm to obtain all frequent itemsets (SBGSets). Finally, we convert SBGSets to sequences, and use definition of patterns to get frequent patterns.

For the case above, "The X is big", "The X is" or "X big" etc. are all patterns. If their frequencies in the SBGSet corpus for extracting patterns are over minimum support, they will be in the set of frequent patterns.

5.4 Selecting and Scoring Patterns

We can score patterns and assume that the word sequences extracted by good patterns are more likely to be product features.

However, although ranking is important for performance, it is not the focus in this paper. So we use simple scheme to select and score patterns. In our experiments, we only consider patterns with the form "$word_1$ X $word_2$", where X is a variable. The score of a pattern p is defined as: $\frac{the\ frequency\ that\ p\ extracts\ seed\ product\ features}{the\ frequency\ that\ p\ extract\ frequent\ product\ features}$.

5.5 Extracting Product Features with Patterns

Suppose that we have the pattern "The X is", two examples are shown in Table 5. It should be pointed out that each sentence is represented as a SBGSet during processing, but we use the original sentence in Table 5 for clarity.

Table 5. Two product features extracted by the pattern "The X is"

The keyboard is big. $\Downarrow$ X is instantiated as "keyboard"
The quality of signal is bad. $\Downarrow$ X is instantiated as "quality of signal"

In our experiments, we assume that a variable is composed of at most three words, which is reasonable since product features are single words or short phrases. After instantiating all the variables in patterns when going through the corpus, we use the same filtering rules in section 4.3 and exclude the frequent product features, finally we obtain the set of infrequent product features.

Since the number of infrequent product features is normally much larger than frequent ones, we use the simple scheme to rank them in order to extract the ones with high scores. We define the score of an infrequent product feature ip is:

$$the\ sum\ of\ scores\ of\ patterns\ that\ can\ extract\ ip.$$

6 Evaluation

6.1 Experiment Setting

In our experiment, we set minimum support to 0.1% (ratio) for Apriori algorithm in extracting frequent product feature for both corpora; in extracting patterns, we set minimum support to 10 (frequency) for the mobile phone corpus and 3 (frequency) the digital camera corpus.

For infrequent product features, if the number of extracted ones are larger than 500, we only keep the top 500 according to the scores defined in section 5.5.

6.2 Evaluation

Given a set of extracted product features E and a gold standard G (a set of product features manually labeled). We use recall (R) and precision (P) to evaluate the extracted product features, $R = \frac{\sum_{x \in E \cap G} Freq(x)}{\sum_{x \in G} Freq(x)}$ and $P = \frac{\sum_{x \in E \cap G} Freq(x)}{\sum_{x \in E} Freq(x)}$, where $Freq(x)$ is the number of the feature x in the corpus.

6.3 The Corpora

We download product reviews on mobile phones and digital cameras from *www.taobao.com*. We use ICTCLAS[2] package to perform word segmentation and POS tagging, during which Specification for Corpus Processing at Peking University in [14] is adopted. Some statistics of two corpora are shown in the Table 6.

Table 6. Statistics of two corpora

Product	Phone	Camera
Corpus size	2396KB	403KB
No. of annotated features	167	454

6.4 Test Sets

Since the corpora are large, it is costly to annotate them in the corpus directly. Instead, after word segmentation and POS tagging, we extract all the word sequences according to the filtering rules in section 4.3, and then perform the annotation. The size of two test sets (two sets of product features) is shown in the last row of Table 6.

6.5 Results of Experiments

The precision in our experiments is lower than some work in English [1,3,6], which is explained from two aspects:

1. In this paper, we pay more attention on the representation of short sequences which can be used to represent product features and patterns, and we do not use any

[2] `http://www.ictclas.cn`

ranking schemes (meronym relationship, feature-opinion association etc.) in our experiments. There ranking schemes[3] help to find a more compact set of product features or patterns.

2. Since we use no Chinese parser for our experiments, we can not obtain noun phrases from a parser. To cover more product features, we allow verbs and other POSs in product features, so the size of candidate product features is large and the precision is low.

Table 7. Results of extracting product features

Product	Mobile phone corpus		Digital camera corpus	
Size of annotated product features	167		454	
Extracted frequent product features	No. of correct	67	No. of correct	146
	No. of Extracted	284	No. of Extracted	630
	Precision	0.4547	Precision	0.4097
	Recall	0.9117	Recall	0.6845
Extracted infrequent product features	No. of correct	37	No. of correct	46
	No. of Extracted	500	No. of Extracted	500
	Precision	0.1249	Precision	0.1557
	Recall	0.0412	Recall	0.0307
All extracted product features	No. of correct	104	No. of correct	192
	No. of Extracted	784	No. of Extracted	1130
	Precision	0.4081	Precision	0.3828
	Recall	0.9529	Recall	0.7153

In Table 7, we can see that frequency is a good indicator for product features. For the extracted frequent product features, recall is 0.9117 for mobile reviews and 0.6845 for camera reviews, which show that frequent product features account for the majority of product features in the corpus.

It is encouraging to find that pattern-based method can find many infrequent product features. Although these product features occur less in the corpus and contribute less to recall, the number of them is not small.

Compared with the experiment on mobile reviews, there are more product features in the domain of digital cameras and the corpus is small, which explains the low performance on camera reviews because it is more difficult to extract reliable sequences (product features or patterns) when product features and the contexts for them are diversified and of low frequency.

7 Conclusion and Future Work

In this paper, aiming to extract product features in Chinese reviews on a product, we propose to use a SBGSet (a set of skip-bigrams) to represent a short word sequence,

[3] Of them, meronym (part of) relationship is most useful. Many high frequent words such as "girl", "old person" can be filtered out by this relationship.

which is of variable length and the common form of product features. The advantage of SBGSet representation is to convert a sequence to a set, thus we can use Apriori algorithm to find frequent itemsets, and then recover the word sequences from the itemsets. Furthermore, we propose a pattern-based method to extract infrequent product features in the corpus, and we also see a pattern as a short sequence that can be represented by a SBGSet.

By adopting SBGSet representation for short word sequences, we can devise filtering rules flexibly and such representation is helpful to wrong word segmentation in Chinese.

In our future work, we will cluster the product features and also make a deep analysis on the association between product features and opinions.

Acknowledgements. This work was finished when the first author worked in the group led by Professor Churen Huang in the Kong Kong Polytechnic University, supported by Joint Supervision Scheme(G-U966) and GRF project(544011,543810). The work is also supported by National High Technology Research and Development Program of China (863 Program) (No.2012AA011101),National Natural Science Foundation of China (No.60973053, No.90920011), and the Specialized Research Fund or the Doctoral Program of Higher Education of China (Grant No.20090001110047).

References

1. Hu, M., Liu, B.: Mining opinion features in customer reviews. In: Proceedings of Nineteeth National Conference on Artificial Intellgience, AAAI 2004 (2004)
2. Hu, M., Liu, B.: Mining and summarizing customer reviews. In: Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD (2004)
3. Hu, M., Liu, B.: Opinion feature extraction using class sequential rules. In: AAAI Spring Symposium on Computational Approaches to Analyzing Weblogs, Palo Alto, USA (2006)
4. Fujii, A., Ishikawa, T.: A system for summarizing and visualizing arguments in subjective documents: Toward supporting decision making. In: Proceedings of the Workshop on Sentiment and Subjectivity in Text, ACL 2006 (2006)
5. Zhao, Y., Qin, B., Hu, S., Liu, T.: Generalizing syntactic structures for product attribute candidate extraction. In: Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the ACL (2007)
6. Popescu, A.M., Etzioni, O.: Extracting product features and opinions from reviews. In: Proceedings of the Human Language Technology Conference and the Conference on Empirical Methods in Natural Language Processing, HLT/EMNLP (2005)
7. Qiu, G., Liu, B., Bu, J., Chen, C.: Expanding domain sentiment lexicon through double propagation. In: International Joint Conference on Artificial Intelligence, IJCAI 2009 (2009)
8. Zhang, L., Liu, B.: Extracting and ranking product features in opinion documents. In: Proceedings of the 23rd International Conference on Computational Linguistics, COLING 2010 (2010)
9. Wang, B., Wang, H.: Bootstrapping both product features and opinion words from chinese customer reviews with cross-inducing. In: Proceedings of IJCNLP 2008 (2008)
10. Su, Q., Xu, X., Guo, H., Wu, X., Zhang, X., Swen, B., Su, Z.: Hidden sentiment association in chinese web opinion mining. In: WWW 2008 (2008)
11. Du, W.F., Tan, S.B.: An iterative reinforcement approach for fine-grained opinion mining. In: Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics (2009)

12. Guthrie, D., Allison, B., Liu, W., Guthrie, L., Wilks, Y.: A closer look at skip-gram modelling. In: Proceedings of the 5th International Conference on Language Resources and Evaluation, LREC 2006 (2006)
13. Agrawal, R., Srikant, R.: Fast algorithm for mining association rules. In: VLDB 1994 (1994)
14. Yu, S., Duan, H., Swen, B., Chang, B.: Specification for corpus processing at peking university: Word segmentation, pos tagging and phonetic notation. Journal of Chinese Language and Computing 13 (2003) (in Chinese)