International Journal of Grid and Distributed Computing
Vol. 9, No. 3 (2016), pp.53-62
http://dx.doi.org/10.14257/ijgdc.2016.9.3.07

# Learning Extraction of Chinese Comparative Sentences for Evaluative Text

Wei Wang, TieJun Zhao, GuoDong Xin

*Department of Computer Science and Technology,*
*Harbin Institute of Technology,*
*Harbin, China*
{wangwei}@hitwh.edu.cn, {tjzhao, gdxin }@hit.edu.cn

***Abstract***

*With the prevalence of Web 2.0, people increasingly prefer to express opinions and exchange information through CGM (consumer-generated media), such as blog, Internet forum and etc. Many studies pay attention to extract and analysis user opinions in consumer reviews. This paper studies how to automatically extract Chinese comparative sentences from consumer reviews. At first, the paper describes a method for solving the class imbalance problem of comparatives and non-comparatives in review data. Then we built a support vector machine learning model to classify comparatives and non-comparatives into different group on a balanced dataset. Experiments were conducted on consumer-generated product reviews, including 9600 sentences, of which 1,624 (16.92% of the total) were comparisons. Experiments show an overall F-score of 87.26%, which presents the effectiveness of the proposed approach.*

***Keywords****: comparative sentence, comparative keyword, frequent sequence, machine learning*

## 1. Introduction

In the age of Web 2.0, the way that people express their point of views has changed. They can express view on hot social issues in forums/blogs, and post product review at online shopping sites. These opinions are precious resources with regard to commercial operations and government decisions. Nowadays, if a person wants to choose a product from multiple candidate products, he/she can look through a large number of product reviews on Web that were given by previous customers. For the managers of a business, if they want to know real feedback from existing customers about their products and services as well as those of their competitors, they could find the user-generated content very useful, which can give them relevant information. Due to the importance of customers' review and market demand, user opinion mining has drawn the attention of many researchers in the field of machine learning and data mining [1-3].

Many of existing studies have acquired opinions about a single product (e.g. “The engine performance of car X is bad/good”). But those studies cannot reveal the comparison between this product and others. To solve it, we focus on comparative opinions such as “The engine performance of A is better than B”. Compared to the opinions about a single product, comparative opinions express relations of similarities or differences among several competitive products. Given by the customers who have used several similar products, comparative opinions are more persuasive and objective. Thus comparative opinions are more convincing than direct opinions [4-6]. Analyzing those comparative opinions can help potential customers to make a more informed choice from many candidate products, and help businessmen to gain competitive insight, which plays a significant role in designing new products and developing market plan.

This paper addresses the problem of extracting comparative sentences from consumer reviews. Our final goal is to establish an automated comparative opinion discovery and summarization system. The system is composed by the following tasks: (1) Identify comparative sentences from review documents. (2) Extract comparative relations from each recognized comparative sentence, i.e. determining what are compared. (3) Sentiment analysis, i.e. determining which objects are preferred by the consumers. (4) Ranking or summarizing compared products according to their sentiment score. In this paper, we aim to study the first task.

To extract comparative sentences, we partition our work into two subtasks: (1) Balance the corpus to solve the problem of class imbalance of categorical datum. (2) Identify comparative sentences from balanced dataset using machine learning technique. In review texts, the distribution between comparative and non-comparative sentences is imbalanced, the number of comparative sentences(CS) is far less than non-comparative sentences(NCS) which may impact on performance of classification system[7]. The first subtask is to address this problem. In the second subtask, we utilize support vector machines (SVM) learning model to classify a Chinese sentence into one of two classes, “comparative” or “non-comparative”.

The remainder of this paper is organized as follows: Section 2 discusses the related work in recognizing comparative sentences. Section 3 states feature selection. Section 4 describes classification learning. Section 5 conducts some experiments and evaluation. Section 6 concludes our study and discusses future direction.

## 2. Related Work

Linguistics and computational linguistics are both related to our study. Researches in linguistics are primarily concerned with syntax and semantics of comparisons, rather than computational identifying technology. Xia investigated comparative category, and sub-category [8]. Chen and Zhou explored various comparative constructs in modern Chinese [9]. They first classified comparative structures into different categories, and then arranged their grammatical items with appropriate order from the linguistic perspective. Che discussed semantic classes of comparisons and syntactic mark words [10].

In computational linguistics, machine learning and pattern match approach are two popular methods for identifying comparisons. Past experiments show machine learning has a better performance in comparative sentence identification. Jindal and Liu mined comparative information between products based on sequential rule mining with continuous Part-of-Speech sequence within the radius of 3 of each keyword. The sequential rules are then used as features of machine learning [4]. Park and Blake have investigated comparisons in scientific articles. They trained three different classifiers using the dependency syntactic features[11]. Compared with machine learning, pattern match is an unsupervised learning and pattern database is difficult to contain all of patterns. Song et al. manually constructed a Chinese pattern database and applied it to mine comparative sentences [12].

Our work is also connected with opinion mining and sentiment analysis, since a number of comparative sentences express customer opinions. We also investigate and refer to some studies about opinion mining [1-3,5,6].

## 3. Feature Representations

### 3.1. Feature Sets 1: Term Features

Some terms, which frequently appear in one class, but hardly appear in the other class, are useful for discriminating different classes. Information gain can measure the effectiveness of a term in classifying the training data [13]. Concretely, it measures the decrease in entropy when a term is present vs. absent in a document. Let $C=\{C_i\}_{i=1}^{m}$ denote the set of classes in the target space. The information gain of a term $T$, relative to a classification system $C$, is defined as:

$$\begin{aligned} IG(C,T) &= \mathrm{Entropy}(C) - Entropy(C \mid T) \\ &= -\sum_{i=1}^{m} P(C_i)\log_2 P(C_i) \\ &\quad + P(t)\sum_{i=1}^{m} P(C_i \mid t)\log_2 P(C_i \mid t) \\ &\quad + P(\bar{t})\sum_{i=1}^{m} P(C_i \mid \bar{t})\log_2 P(C_i \mid \bar{t}) \end{aligned} \tag{1}$$

Where $t$ denotes term $T$ appeared, whereas $\bar{t}$ denotes term $T$ does not appeared. The first term in the formula (1) is the entropy of the original classification system. The second term is the expected value of the conditional entropy of system given a term. For each term in the training data we computed information gain, and selected those terms whose gain value is higher than specified threshold. The algorithm of term features selection is shown in algorithmic 1. Where $TS$ denotes the set of all terms in the training corpus. $TS_{goal}$ denotes the set of target terms with high gain value. Finally, 139 terms were collected into $TS_{goal}$.

```
Algorithm 1: Term Extraction Algorithm
Input:  Training Corpus $S$, Information Gain Threshold $\alpha$
Output:  Target Term Set $TS_{goal}$
Method:
1.  $TS \leftarrow \phi, TS_{goal} \leftarrow \phi$
2.  For each sentence $s \in S$ do
3.  For each term $t \in s$ do
4.  If $t \notin stop\ words$ then
5.      $TS \leftarrow t$;
6.    Endfor
7.  Endfor
8.  For each term $t \in TS$ do
9.  Calculate $IG(t)$;
10. If $IG(t) > \alpha$ then
11. $TS_{goal} \leftarrow t$;
12.   t.count++ ;
13. Endfor
14. Return $TS_{goal}$;
```

## 3.2. Feature Sets 2: Comparative Keywords

Some words can express the comparative relationship between entities or introduce comparative objects, which play an important role in identifying comparative sentences. To find such keywords, we group the comparative sentences into five types and find comparative keywords for each type as follows.

**Table 1. The Classification of Comparisons**

| | Type | Keyword |
|---|---|---|
| 1 | Equality | 相同,相似(same, similar) |
| 2 | Difference | 不同(different) |
| 3 | Superlative | 最(most) |
| 4 | Greater or Lesser | 比( than) |
| 5 | Implicit comparison | Words-and-POS-tags sequence |

We can find explicit keywords for the sentences in first 4 types. The basic approach is synonym extension. We construct an initial set of keywords $CK_{init}$ based on linguistics literature [9,10]. However, linguistics-based $CK_{init}$ is not necessarily to cover all real comparative expressions. Therefore, we expand $CK_{init}$ into the set of candidate keywords $CK_{can}$ by synonym extension as follows:

$$CK_{can} = CK_{init} \cup \{x \mid x \in the \text{ synonym } set\ of\ CK_{init}\} \tag{2}$$

Given a collection of CS, we remove those words whose frequency value is less than the predefined threshold from $CK_{can}$. We can't find any explicit keyword for the sentences in type 5, such as:

*Example1. "X 有摄像功能，而 Y 没有摄像功能(X has camera function, but Y does not have.)"*

Such sentences commonly compose of two short sentences that express contrast. Hence, we build Words-and-POS-tags sequence which act as keyword for them. For example "<NN, 有, 而，NN，没有>(<NN has but NN does not have>)" is a Words-and-POS-tags sequence extracted from example 1, where NN denotes the POS( Part of Speech) tag of noun. In addition, some phrases and idioms that indicate comparisons are regarded as the keywords. Therefore, a comparative lexicon is defined as:

$$\begin{aligned} Comparative\ Lexicon = {} & CK_{can} \cup \\ & \{phrases\ e.g.\ compared\ with,\ different\ from...\} \cup \\ & \{idioms\ e.g.\ entirely\ same,\ etc\} \cup \\ & \{words\ and\ POS\ tags\ sequences\} \end{aligned} \tag{3}$$

There are a total of 106 words and 30 sequences in our comparative lexicon. After a comparative lexicon has been built, we use it to balance our corpus. Concretely, if a sentence contains one or more comparative keyword, the sentence is added into the balanced corpus. Our balanced corpus has comparative sentences of 41.68%, non-comparatives of 58.32%.

### 3.3. Feature Sets 3: Frequent Sequences

Comparative sentences have special language patterns different from non-comparatives, which can be used as the features of machine learning. We automatically extract frequent sequences by mining sequential patterns. Some infrequent patterns are manually built.

***A. Class sequence rule***

Sequential pattern mining (SPM), which extracts all sequential patterns from a sequence database, is an important data mining task. A sequential pattern, also called frequent sequence, is a subsequence whose support exceeds a predefined minimal support threshold[14].

**Sequence:** Let $I = \{i_1, i_2, ..., i_n\}$ be a set of all distinct items. An itemset is a non-empty subset of items. A sequence $s$ is an ordered list of itemsets $< s_1, s_2, ..., s_m >$, where $s_i$ is a itemset, and $s_i$ is denoted by $\{x_1, x_2, ..., x_l\}$, where $x_j$ is an item. A sequence $s_1 =< a_1, a_2, ..., a_n >$ is contained in another sequence $s_2 =< b_1, b_2, ..., b_m >$, if there exist integer $1 \le i_1 \prec i_2, ... \prec i_n \le m$ such that $a_1 \subseteq b_{i_1}, a_2 \subseteq b_{i_2}, ..., a_n \subseteq b_{i_n}$.

**Sequence database:** Let $S$ be a sequence database. Each sequence $s_i \in S$ is attached a class label $y_i \in Y$. $Y$ is the set of all classes. In our task, $Y = \{comparative, non-comparative\}$. The sequence database $S$ is represented with $S = \{(s_1, y_1), (s_2, y_2), ...(s_n, y_n)\}$, where $s_i$ is a sequence in $S$, $y_i \in Y$ is its class.

**Class sequence rule(CSR):** A **class sequence rule** (CSR) is an implication $X \to y$, where $X$ is a sequence, $y \in Y$. A data instance $(s_i, y_i)$ in $S$ is called to support a CSR if $s_i$ contain $X$. A data instance $(s_i, y_i)$ is called to satisfy a CSR if $s_i$ contain $X$ and $y_i = Y$. The *support* of the rule is defined as the fraction of total instances in $S$ that satisfies the rule.

$$Support(CSR) = \frac{\text{instance number of satisfy rule}}{\text{Total number of instances}} \tag{4}$$

The *confidence* of the rule is defined as the proportion of instances in $S$ that supports the rule also satisfies the rule.

$$Confidence(CSR) = \frac{\text{instance number of satisfy rule}}{\text{instance number of supportrule}} \tag{5}$$

***B. Mining indicative pattern***

In order to mine *CSR*, we firstly transform corpus into a set of sequences. Each sentence in training set is broken up into several clauses by punctuation. We find all clauses having keywords and perform Chinese word segmentation and POS tagging for them. For each clause that comprises at least one keyword, we use actual word of each keyword as an item, for other words, we use the POS of each word as an item to produce a sequence. In order to adapt to various expression of comparative, we use POS tags of some words to form sequences. Each sequence is attached a class tag according to whether the sentence is a comparative or non-comparative sentence. We use $S_C$ to store the set of comparative sentences, $S_k$ to store the set of keyword, and $SEQ_C$ to store the set of built sequences with comparative class.

**Algorithm 2**: constructing sequence data

```
Input: $S_C$, $S_{NC}$, $S_k$
Output: $SEQ_C$, $SEQ_{NC}$
Method:
  1. $SEQ_C = \phi, SEQ_{NC} = \phi, C = \phi$
  2. for each $s_i \in S_C$ do
  3.    breaking up $s_i$ into a set of clause $C_i$
  4.     for each $c_{ij} \in C_i$
  5.        for each $k_l \in S_k$ do
  6.          if $c_{ij}$ contains $k_l$ then
  7.              $C \leftarrow c_{ij}$
  8.          endfor
  9.     endfor
  10. endfor
  11. for each $c_i \in C$
  12.   segmenting word and POS tagging to $c_i$
  13.   for each $word_{ij} \in c_i$
  14.     if $word_{ij} \in S_k$ then
  15.        $word_{ij}$ as an item of $seq_i$
  16.      else
  17.        $POS_{ij}$ as an item of $seq_i$
  18.    endfor
  19.    $seq_i \rightarrow SEQ_C$
  20. endfor
  21. return $SEQ_C$        //  For $S_{NC}$, the same process is performed as $S_C$
```

*Example 2.* "在/P 1.8L/CD 的/DEG 轿车/NN 里/LC ,/PU 花冠/NR 的/DEG 发动机/NN 最/AD 棒/JJ !/ PU (In the 1.8L sedan, the engine of Corolla is the best !)" .

Example 2 contains two clauses, and has keyword "最(best)" in the second clause. So the final sequence for the second clause is:

<{NR}{DEG}{NN}{最}{JJ}>   Comparative

CSR mining algorithms can be produced by simply modifying any of sequence pattern mining algorithms such as GSP, PrefixSpan[14,15]. We use PrefixSpan algorithm to extract all CSR. The output results of the algorithm need to meet the minimum confidence threshold (in our experiment, 70% can work best). The minimum support needs to be set multiple values in our context because some comparative keywords appear very frequently, while some others appear rarely. If setting a single minimum support, it is difficult to capture those patterns generated by infrequency words. In this strategy, each keyword is set a minimum support which is in proportion to word frequency in the training set.

### 3.4. Feature Sets 4: Infrequent Sequences

We cannot find any frequent sequence for some keywords, such as “最(most)” etc. They not only often appear in the comparatives sentences but also frequently appear in the non-comparatives. For such keywords, we use consecutive POS tags within the radius of 2 of each keyword as a sequence. Similarly, each keyword use actual word as a term. In addition, we manually build some sequences for these keywords.

## 4. Classification Learning

In this section, we train a SVM classifier to classify whether a given sentence $s$ is comparative(c = 1) or non-comparative(c = 0). Formally, let $S = \{s_1, s_2, ..., s_M\}$ be a set of sentences in a collection D. We convert each sentence into a feature vector $x \in R^n$ , where $n$ represents the number of features. The feature set is:

$$\begin{aligned} Feature\,Set = \ & \{X \mid X \ is\ term\ \ feature\} \cup \\ & \{Y \mid Y\ is\ comparative\ keyword\} \cup \\ & \{Z \mid Z\ is\ \ frequent\, sequence\} \cup \\ & \{W \mid W\ is\ infrequent\, sequence\} \end{aligned} \tag{6}$$

Specifically, the feature $x_i \in \{0,1\}$ of a sentence corresponds to whether the i-th feature occurs in the sentence. Let $c \in \{0,1\}$ is a class variable such that $c = 1$ represent comparative and $c = 0$ represent non-comparative. The classifier will predict $c_i$ of sentence $s_i$ based on its feature vector $x$ .

## 5. Experimental Evaluation

### 5.1. Data Sets

Our data is composed of the consumer reviews on such products as automobiles, electronics, from task 2 of the fourth Chinese Opinion Analysis Evaluation (COAE2012) [16]. Table 2 shows the distribution of comparative and non-comparative sentences, a total of 9600 sentences, of which 1,624 (16.92% of the total) are comparisons. We use LIBSVM package [17-19] with the RBF kernel to perform classification, the optimal model parameter is gamma = 0.007813 and C = 32.

**Table 2. Number of Sentences in Each Dataset**

| **Data set** | **Comparative Sentences** | **Non-Comparative Sentences** |
|---|---|---|
| electronic products | 811 | 3989 |
| automobile | 813 | 3987 |
| Total | 1624 | 7976 |

First of all, we perform an experiment with keyword searching to balance two classes of data. There are four types of sentences in original corpus, which are shown in Table 3.

**Table 3. The Proportion of Four Types of Sentences**

| | Type | ratio |
|---|---|---|
| 1 | CS with CK | 16.46% |
| 2 | CS without CK(Implicit) | 0.46% |
| 3 | NCS with CK | 23.03% |
| 4 | NCS without CK | 60.05% |

Using keyword searching, we can eliminate most of non-comparative sentences, and get a relatively balance dataset including comparative sentences of 41.68%, non-comparatives of 58.32%. Recall of 97.29% for comparative sentences indicates that most of comparative sentences are included in balanced corpus, i.e., the keywords in comparative lexicon can cover almost all comparative sentences. Table 4 shows the ratio of CS and NCS before and after corpus is balanced.

**Table 4. Sentence Distribution of Different Stages**

| Data set | Comparative Sentences | Non-Comparative Sentences |
|---|---|---|
| Original corpus | 16.92% | 83.08% |
| Balanced corpus | 41.68% | 58.32% |

Then, we conduct some experiment as follows:

To check which features can provide more information to the system, we compare the classification performance with single feature and that with the combination of features. The features include the terms (denoted as TM), the comparative keywords (denoted as CK), the frequent sequences (denoted as FS) and the infrequence sequences (denoted as IFS). As a result, the combination of all features shows the best performance.

Table 5 and Table 6 exhibit the experiment results in automobile and electronic corpus respectively. In each case, the recall value is far less than precision value. For single feature, keyword and frequent sequence gain higher performance than term feature. For combined features, keyword plus term features show similar performance with the combination of sequence features. When all lexical and sequence features are used, the system shows the optimal performance, F-score of 83.75% and 90.76%.

**Table 5. The Classification Performance in Automobile Data**

| Features | Automobile | Precision | Recall | F-score |
|---|---|---|---|---|
| KW | Balanced | 0.8758 | 0.6964 | 0.7728 |
| TM | Balanced | 0.7862 | 0.6237 | 0.6938 |
| FS | Balanced | 0.9017 | 0.7129 | 0.7947 |
| IFS | Balanced | 0.9374 | 0.6310 | 0.7534 |
| KW+ TM | Balanced | 0.8868 | 0.7422 | 0.8075 |
| FS+IFS | Balanced | 0.8996 | 0.7340 | 0.8082 |
| KW+ TM+ FS+IFS | Balanced | 0.9001 | 0.7850 | 0.8375 |

**Table 6. The Classification Performance in Electronic Data**

| Features | Electronic | Precision | Recall | F-score |
|---|---|---|---|---|
| KW | Balanced | 0.9064 | 0.8027 | 0.8502 |
| TM | Balanced | 0.8390 | 0.7708 | 0.8026 |
| FS | Balanced | 0.9516 | 0.8200 | 0.8804 |
| IFS | Balanced | 0.9577 | 0.7372 | 0.8317 |
| KW+ TM | Balanced | 0.9206 | 0.8501 | 0.8832 |
| FS+IFS | Balanced | 0.9551 | 0.8296 | 0.8874 |
| KW+ TM+ FS+IFS | Balanced | 0.9512 | 0.8684 | 0.9076 |

## 6. Conclusion and Future Work

In this paper, we designed an approach to identify comparative sentences from customer opinion data. This scheme is composed of two steps: (1) Comparative keyword search is executed to removes a large number of non-comparative sentences so as to obtain a relative balanced dataset between the classes. (2) A support vector machine model is built to determine whether a sentence is a comparative sentence or not. In our study, various linguistic features are introduced, such as terms, comparative keywords, frequent sequences etc. The experiment evaluations demonstrate that our scheme is effective.

In the future, we plan to find more effective features that represent a sentence to further improve the recall of our system. We also plan to extract comparative relations from the identified comparative sentences.

## Acknowledgement

The work of this paper is funded by the project of National Natural Science Foundation of China (No.61173073, No.61172099, No.61073130, No. 61100093, No. 61272384), the Special Project of International Science and Technology Cooperation of China (No. 2014DFA11350).

## References

[1] B. Pang, L. Lee, “Opinion mining and sentiment analysis” , Foundations and Trends in Information Retrieval, vol. 2, no.1-2, (2008), pp.1-135
[2] S.M Kim, E. Hovy, “Automatic identification of pro and con reasons in online reviews”, In Proceedings of the COLING/ACL on Main Conference Poster Sessions, Association for Computational Linguistics, (2006), pp.483-490
[3] B. Liu, M. Hu, J. Cheng, “Opinion observer: analyzing and comparing opinions on the web”, In Proceedings of World-Wide Web Conference(WWW'05) ,(2005)
[4] N. Jindal and B. Liu, “Identifying Comparative Sentences in Text Documents”, In Proceedings of SIGIR’06, (2006) , pp.244-251.
[5] M. Ganapathibhotla and B. Liu, “Mining Opinions in Comparative Sentences”, In Proceedings of Coling’08, (2008), pp.18-22.
[6] K. Q. Xu, S. S. Y. Liao, J. X. Li and Y. X. Song, “Mining Comparative Opinions from Customer Reviews for Competitive Intelligence”. Decision Support Systems, vol. 50, no 4, (2011), pp. 743-754.
[7] N.Japkowicz, S.Stephen, “The class imbalance problem: a systematic study”, Intell. Data Anal, vol. 6, no.5, (2002), pp.429-449
[8] Q. Xia, “A Review on Studies of Comparative Sentences of Chinese”, Chinese Language Learning, no 2, (2009), pp. 58-64.
[9] J. Chen, and X. Zhou, “The Selection and Arrangement of Grammatical Items concerning Comparative Sentences”, Language Teaching and Research, no.2, (2005) , pp.22-33.
[10] J. Che, “A Brief Analysis of Comparative Sentences in Modern Chinese”, Journal of Hubei Normal University (Philosophy and Social Science), vol. 25, no.3, (2005), pp. 60-63.

[11] D. Park, , and C. Blake, “Identifying Comparative Claim Sentences in Full-Text Scientific Articles”, In Proceedings of ACL’12, (2012), pp.1-9.
[12] R. Song, H. F. Lin, and F. Chang, “Chinese Comparative Sentences Identification and Comparative Relations Extraction”, Journal of Chinese Information Processing, vol. 23, no. 2, (2009), pp.102-107.
[13] T. Mitchell, “I Machine Learning”, McGraw Hill, (1997) , pp.55-60
[14] J. Pei, H. Pinto, Q. Chen, J. Han, B. Mortazavi-Asl, U. Dayal, and M. Hsu. “PrefixSpan: Mining sequential patterns efficiently by prefix-projected pattern growth”, In Proceedings of IEEE International Conference on Data Engingeering (ICDE-2001), (2001).
[15] B. Liu, “Web Data Mining: Exploring Hyperlinks, Contents, and Usage Data(Second Edition) ”, Springer , (2011).
[16] K. Liu, S. Wang, X. Liao and H. Xu, “Overview of Chinese Opinion Analysis Evaluation 2012”, In: Proceedings of the 4st Chinese Opinion Analysis Evaluation, NanChang, China: The Professional Committee of Information Retrieval, (2012), pp.1-32.
[17] C.C. Chang, C.J.Lin, “Libsvm: a library for support vector machines”, Trans Intell Syst Technol , vol. 2, no. 3, (2011), pp.27
[18] C, Cortes, V. Vapnik, Support-vector networks. Mach. Learn, vol. 20, no. 3, (1995) , pp.273-297
S. Yang and Y. Ko, “Finding Relevant Features for Korean Comparative Sentence Extraction”, Pattern Recognition Letters, vol. 32, no 2, (2011), pp.293-296.
[19] C.J. Burges, “A tutorial on support vector machines for pattern recognition”, Data Min Knowl Discov, vol. 2, no 2, (1998) , pp.121-167