Design and Implementation of an Intelligent Automatic Question Answering System Based on Data Mining

Zhe Qu[1] and Qin Wang[2]

[1] School of Computer Science, The University of Birmingham,
B15 2TT Birmingham, UK
[2] School of Information Science and Engineering,
University of Jinan, 250022 Jinan, China
alicequz@gmail.com, ise_wangq@ujn.edu.cn

Abstract. This paper applies data mining algorithms into intelligent question answering system, proposes a question answering scheme based on an improved data mining algorithms and achieves an efficient intelligent question answering system.

Keywords: data mining, association rules, question answering system.

1 Introduction

With the development and application of Internet, intelligent information technology is developing rapidly[1]. The network has become an effective tool for people to obtain information, and answering questions face to face is not the mainest learning mode any more[2]. Web-based self-study has become another major learning styles to students. Then plain text keyword queries online question answering system came out. This system has already had the initial feature of intelligent question answering--the database have questions with their corresponding answers and students can enter the keywords of the problem to search. If the entered keyword is not precise enough, it may find many unrelated questions and their answers.

As the sharp increase in the demand of students to obtain knowledge, and the constantly enriching of the network content, it is necessary to design more efficient question answering systems, one of which is intelligent question answering system based data mining[3].

The purpose of this paper is to use Apriori algorithm[4] reform the existing intelligent question answering system, which can be through machine processing automatically understand the user's natural language questions and automatically returns the answer, which makes users more easily use the system, and the system returns the answer with a higher effectiveness. In short, the answer is more effective, less time queries, manage more convenient and better help students to learn, improve network quality and efficiency of teaching.

In the current model of online teaching, the design and development of an intelligent question answering system has some meaning: (1) In network, timely and effective access to solutions is the basic needs for students learning. (2) Students can

Y. Tan, Y. Shi, and Z. Ji (Eds.): ICSI 2012, Part II, LNCS 7332, pp. 137–146, 2012.

easily and freely ask questions and promptly resolve problems, so that the online teaching really play the role to aid learning. This paper using data mining method proposed an intelligent question answering system scheme.

2 Research and Implement of Core Algorithms

2.1 The Apriori Algorithm and Its Improvement

In 1993, Agrawal et al-who are researchers of IBM's Almaden Research Center-published a paper first proposed the concept of association rules[3], has given an algorithm for mining association rules, more of this algorithm is to show that mining association rules is feasible in theory, but clearly there is a great lack of efficiency, thus it is difficult to do real-world practical applications. Until 1994, also by Agrawal et al's paper which was published in 1994 put forward a well-known algorithm-Apriori algorithm[4], this algorithm became the most classic algorithm of all the association rules mining algorithms, a large number of follow-up study revolved around improving the efficiency of the algorithm. Relative to Agrawal et al in 1993 first proposed algorithm, Apriori algorithm achieves a qualitative leap in efficiency, so that the theory of mining association rules can actually be applied to the real world.

The Basic Idea of Apriori Algorithm

Apriori property[2]: all the non-empty subsets of frequent itemsets must be frequent. Apriori property based on the following observation: by definition, if the itemset *I* does not meet the minimum support min_sup, then *I* is not frequent, i.e.

$$P(I) < \text{min\_sup} \tag{1}$$

If item *A* added into *I*, the result itemsets (i.e., $I \cup A$) not possible appear more frequently than *I*. So $I \cup A$ is not frequent, i.e.

$$P(I \cup A) < \text{min\_sup} \tag{2}$$

The basic idea of Apriori algorithm can be described as follows:

Algorithm: Apriori finds the frequent itemsets according to the layer by layer iterative generated by the candidate set.

Input: transaction database D; minimum support min_sup.

Output: frequent itemsets in D.

Method:

```
$L_1$=find_frequent_1-itemsets(D);   //frequent 1-itemset
for(k=2;$L_{k-1}\neq\Phi$;k++){
  $C_k$=apriori-gen($L_{k-1}$,min_sup);   //new candidate sets
  for each transactions $t\in$D{
    $C_t$=subset($C_k$,t);   // candidate sets included in
    transaction t
    for each candidates $c \in C_t$
         c.count++;
    }
$L_k$={$c \in C_k$ | c.count$\geq$minsup}
}
return $L=\cup_k L_k$
```

Apriori algorithm using a layer by layer search iteration method, k-itemset used to explore the (k +1)-itemset. First, generate all the frequent 1-itemsets, and then on this basis generate in turn frequent 2-itemsets, frequent 3-itemsets......, until it can not find the frequent itemsets at all. It needs to scan the database at each time when generating a frequent itemset.

The Weaknesses and Improvement of Apriori Algorithm

Using the Apriori algorithm for association rule mining, can more effectively generate association rules, but there are two shortcomings:

One is the algorithm has problems in efficiency. Main reason is that it has to scan the database for too many times, looking for every k-frequent itemsets (k = 1, 2,..., K) need to scan the database once, totally scanning K times. In addition, when the mode is too long, the large numbers of generated candidate itemsets are unacceptable. So when the database or K is too large, time-consuming of the algorithm is too long and the algorithm can not be completed. Therefore, scalability of the algorithm is not strong and difficult to promote.

Another is the algorithm generates too many false (redundant) rules. When the data warehouse is too large or the support and confidence is too low, it will generate too many rules that the user is difficult to make an artificial distinction and judgment between those rules, making it difficult to find really useful knowledge to the users.

Aimed to the above two disadvantages, this paper presents an improved association rules algorithm, greatly improving the mining efficiency, and to some extent, solved the redundancy problem. When using Apriori mines association rules, in order to determine the count of each candidate item, it needs to repeatedly scan the transaction database. This paper aims to reduce the number of scanning, put forward an improved algorithm to get the frequent itemsets, the algorithm still based on support and confidence. Algorithm according to: (1) All candidate itemsets C_k are supersets of the item C_{k-1}; (2) if a transaction does not contain candidate itemset, then delete the transaction will not affect the generation of frequent itemsets.

Algorithm process: to add a flag *bool* to each item in the transaction database, default is false. In each calculating the number of occurrences of candidate itemset C_k, if the transaction does not include any particular item of C_k, the value of the flag set to true. If includes, the value is set to false, and give the count plus 1, and the flag does not change any more. If finally the flag is true, that the transaction does not contain any item of the candidate itemsets, then remove it from the transaction database. To compute the support of candidate itemset, the number of records involved will be less than the actual number of records in the transaction database and with the k value increases, the difference has continued to grow, thus effectively reducing the count of computing candidate itemsets and improve the overall efficiency of the algorithm.

Improved Apriori algorithm is as follows:

Input: transaction database D; minimum support min_sup

Output: the frequent itemsets L in D

```
L1=find-frequent_1_itemsets(D);
For(k=2; $L_{k-1}\neq\Phi$; k++){
    $C_k$ =apriori_gen $L_{k-1}$, min_sup);
    for each transaction $t\in D${//scan D for count
    $C_t$ =subset($C_k$,t);//get the subsets of t that are
candidates
    If  t.bool=false then { //set bool sign
    If  $c\notin C_t$  then {
    t. bool =true;
    break;}
    else  c.count++;}
else  if  $c\in C_t$ then{ c.count++;
    bool =false }
    } //for each
    if t. bool =true then
    delete t from transaction
    $L_k=\{c\in C_k \mid c.count\geq min\_sup\}$
}
return $L=\cup_k L_k$;
```

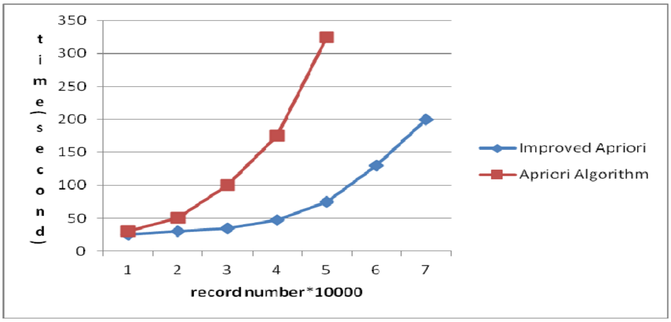


Fig. 1. Contrast of Apriori Algorithm and the improved Apriori Algorithm

Figure 1 shows the difference between the Apriori algorithm and the improved Apriori algorithm of this paper in execution time with the increase in the number of records. From the experimental results can be seen, the impoved algorithm compared with the Apriori algorithm has much higher efficiency in the case of requring low number of records and low support degree. And because they have found the same rules, the accuracy of rules found of improved algorithm are the same with the Apriori algorithm.

The Application and Implementation of Apriori Algorithm

This paper uses the improved association rules algorithm to to compute frequent itemsets in questions, which composed of words that frequently appear together.

Like the majority of text-based analysis, association analysis first need to do the segmentation of the text, then call association rule mining algorithms. Make a words association table using the generated association rules, through the association table to calculate the relevance (support and confidence) between words. As follows:

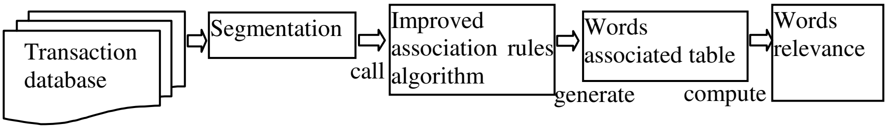


Fig. 2. The process of Apriori Algorithm

All texts will be considered as a transaction in a Q&A table, each text contains a collection of words as a set of transaction, then the transaction can be expressed as:

{Text, Word 1, Word 2, Word 3,..., Word n}
Such as:

What is the CPU?	cpu			
What are the hardware supports of virtual memory?	hardware	support	virtual	memory
What are the key technologies of virtual memory?	key	technol-ogies	virtual	memory

As the traditional association rule mining algorithm, the word-based text analysis also includes two major steps:

Step 1: Mining frequent co-occurrence words, i.e. frequent itemsets.

Step 2: Generating association rules between words according to frequent itemsets.

The mining association rules result are as the form of (no, front, rear, S, C), its meaning is (the number of generated association rules, association rules FRONT, association rules REAR, support, confidence), in which the association rules need through the combination of FRONT and REAR. Partial results are shown in Table 1.

Table 1. Association rules table based on words

No	*FRONT*	*REAR*	*S*	*C*
1	System	windows	0.0183	0.3077
2	Computer	hardware	0.0137	0.5
3	Memory	virtual	0.0137	0.75

If not satisfied with the results, it can adjust the support and confidence values, re-run the training process until satisfied.

2.2 Searching Answer Algorithm

Let the users' input question after segmentation be the Set Q,

$$Q = \{q_1, q_2, ..., q_n\} \tag{3}$$

where q_1, q_2 ... is the separate word after segmentation.

Let the separate words collection after segmenting the existed question sentences be Set A_n,

$$A_1 = \{a_{11}, a_{12}, a_{13}, ...\} \tag{4}$$

$$A_2 = \{a_{21}, a_{22}, a_{23}, ...\} \tag{5}$$

$$\cdots\cdots$$

$$A_n = \{a_{n1}, a_{n2}, a_{n3}, ...\} \tag{6}$$

where a_{ij} is the separate word segmented from the existed question sentences.

Let frequent 2-itemset be Set S_2,

$$S_2 = \{s_{21}, s_{22}, s_{23}, ...\} \tag{7}$$

where S_{2i} is each association rule in frequent 2-itemset.

Let frequent 3-itemset be Set S_3,

$$S_3 = \{s_{31}, s_{32}, s_{33}, ...\} \tag{8}$$

where S_{3i} is each association rule in frequent 3-itemset.

Let the similarity of question matching be *sim*.

According to the Set above, compute the similarity of users' input questions and the existed question which stored in database, the algorithm is as follows:

First intersect the set Q and the set A_i, and the results are the match of users' input questions and the existed question.

Let the number of items of the intersection above be n_1, then

$$simi_1 = n_1 \bullet k_1 \tag{9}$$

where k_1 is the similarity weight. Since this computation is simply words matching, so just give a small value to k_1.

Combine every two items in the set Q, and then seek the intersection of set S_2 and Q.

Let the number of items of the intersection above be n_2, then

$$simi_2 = n_2 \bullet k_2 \tag{10}$$

where k_2 is the similarity weight.

Similarly, after the intersection computation of Q and S_3, get the similarity:

$$simi_3 = n_3 \bullet k_3 \tag{11}$$

As the frequent 3-itemset match could show high similarity, so

$$k_1 < k_2 < k_3 \tag{12}$$

Now sum up the three similarities,

$$sim = simi_1 + simi_2 + simi_3 \tag{13}$$

could get the similarity value of user input question with each question existed in database. The answer of the user input question is the answer of the biggest similarity corresponding question.

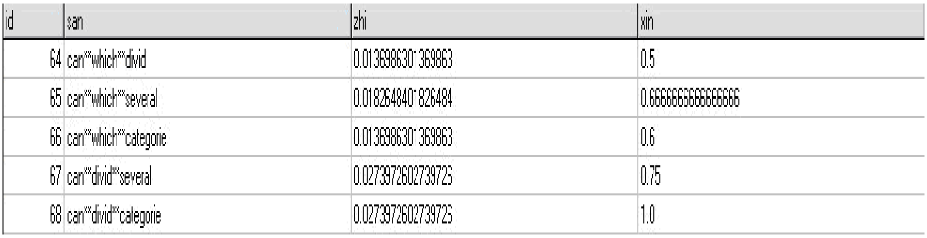
To sum up, this searching answer algorithm uses frequent itemsets other than the whole Q&A to search answers to improve the efficiency, and searching frequent itemsets improved accuracy as well.

2.3 Segmentation

As the nature of English, words in sentences are separated by spaces, so use space as mark to segment English sentences for the method of the English segmentation. At the same time, cut off adverbs, punctuations and other characters etc. which do not make sense, to get separate words.

3 Database Design

In a high-performance system, the database design is essential, which directly determines the efficiency of the system. This system includes six main tables, for storing QA pairs, questions and their words segmentation, 1-itemset, 2-itemset, 3-itemset and similar words respectively. Among them, the n-itemset (n=1,2,3) are the generated Association rules according to the questions.



id	san	zhi	xin
64	can**which**divid	0.0136986301369863	0.5
65	can**which**several	0.0182648401826484	0.6666666666666666
66	can**which**categorie	0.0136986301369863	0.6
67	can**divid**several	0.0273972602739726	0.75
68	can**divid**categorie	0.0273972602739726	1.0

Fig. 3. 3-itemset table

4 System Design and Implementation

The system mainly has three modules respectively for user, administrator and the system itself.

Firstly, the original database has plenty of questions and answers information; the users could input the questions in the text field, then the system could get the sentences and segment them into words. The Searching Answer Algorithm could work on the words and the database to compute the similarity. Sort according to similarity values then obtain in turn the questions and answers with high similarity value. The answer with highest similarity value can be considered to be the direct simple answer, others with high value can be considered to be related answers.

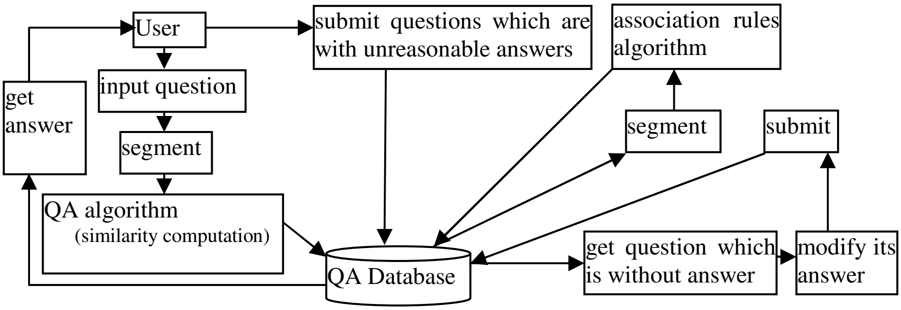
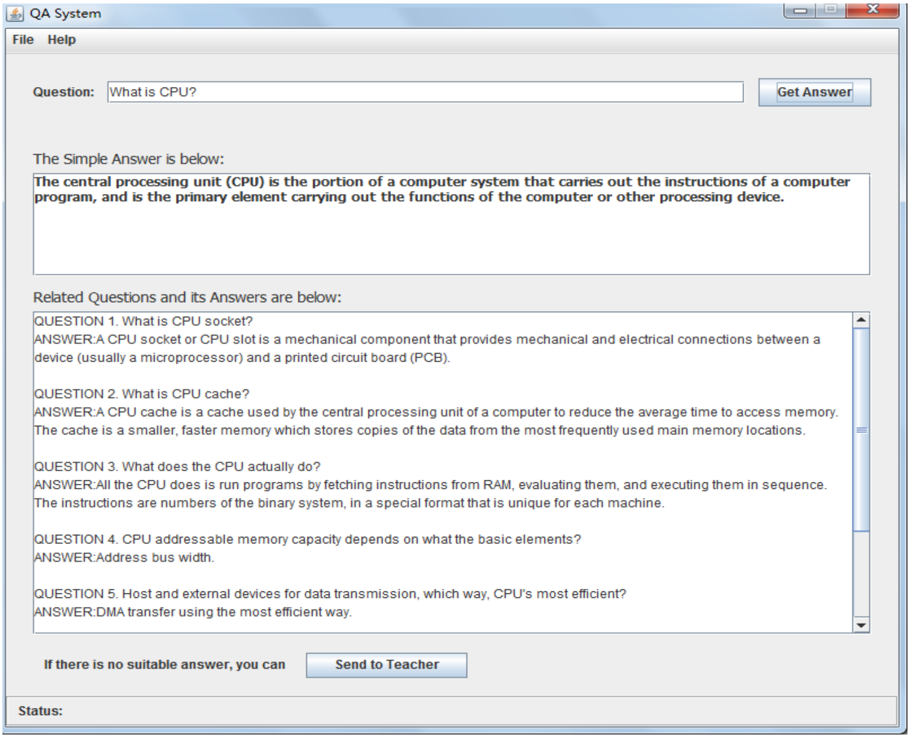

User
submit questions which are with unreasonable answers
association rules algorithm
get answer
input question
segment
segment
submit
QA algorithm (similarity computation)
QA Database
get question which is without answer
modify its answer

Fig. 4. The System Overall Design Flow Chart



Question: What is CPU?

The Simple Answer is below:

The central processing unit (CPU) is the portion of a computer system that carries out the instructions of a computer program, and is the primary element carrying out the functions of the computer or other processing device.

Related Questions and its Answers are below:

QUESTION 1. What is CPU socket?
ANSWER:A CPU socket or CPU slot is a mechanical component that provides mechanical and electrical connections between a device (usually a microprocessor) and a printed circuit board (PCB).

QUESTION 2. What is CPU cache?
ANSWER:A CPU cache is a cache used by the central processing unit of a computer to reduce the average time to access memory. The cache is a smaller, faster memory which stores copies of the data from the most frequently used main memory locations.

QUESTION 3. What does the CPU actually do?
ANSWER:All the CPU does is run programs by fetching instructions from RAM, evaluating them, and executing them in sequence. The instructions are numbers of the binary system, in a special format that is unique for each machine.

QUESTION 4. CPU addressable memory capacity depends on what the basic elements?
ANSWER:Address bus width.

QUESTION 5. Host and external devices for data transmission, which way, CPU's most efficient?
ANSWER:DMA transfer using the most efficient way.

If there is no suitable answer, you can

Fig. 5. The interface of user module

If the user could not get the answer or cannot get the answer they want, they could submit the question to the system and wait for someone to answer it. Then the question stored into the database.

Secondly, for the administrator or the teacher who answers the questions, they could get all the questions and answers from the database and could modify the answers in their interface. If they submit the modified information to the system, these contents will be stored into the database.

Thirdly, for the system, new questions and answers will increase with the use of the system, so the system needs to compute new association rules. Use the current Q&A database to do the segmentation and association rules algorithm to produce new association rules, so that it could compute and get more answers for users.

5 Conclusions

This paper proposed an improved Association Rule Algorithm which improved efficiency by reducing the number of scanning the transaction database. Then design a Searching Answer method based on the association rules which increased the efficiency of the user to get answers. And finally implemented a Question Answering System; the system can get answers quickly and effectively by the obtained association rules and the searching answer method.

References

1. Qu, S., Wang, Q.: Data Warehouse Design for Chinese Intelligent Question Answering System Based on Data Mining. In: Second International Conference on Innovative Computing, Information and Control (ICICIC 2007), pp. 1793–8201 (2007)
2. Qu, S.N., Wang, Q., Zou, Y., et al.: Intelligent Question Answering System Based on Data Mining. Journal of Zhengzhou University (Natural Science Edition) 2, 50–54 (2007)
3. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules. In: Proceedings of the 20th International Conference on Very Large Databases, Santiago, Chile, pp. 487–499 (September 1994)
4. Agrawal, R., Imielinski, T., Swami, A.: Mining association rules between sets of items in large database, pp. 207–216. SIGMOD, Washington, DC (1993)
5. Agrawal, R., Shafer, J.C.: Parallel mining of association rules: design, implementation, and experience. IBM Research Report RJ 10004 (1996)
6. CISL: a computing laboratory, Introduction to Data Mining and Knowledge Discovery (2011), `http://www.scd.ucar.edu/hps/GROUPS/dm/dm.html` (accessed June 11, 2011)
7. Dong, C., Qu, S., Xu, D., Liu, P.: Research on Algorithm of Association rules and its applications in course correlation. Computer Science and Practice (9), 109–112 (2004)
8. Sebastiani, F.: Machine learning in automated text categorization. ACM Computing Surveys, 1–47 (2002)
9. Witten, I.H., Frank, E.: Data mining: practical machine learning tools and techniques, 2nd edn. Elsevier (2005)
10. Han, J., Kamber, M.: Data Mining Concepts and Techniques, 2nd edn. Morgan Kaufmann Publishers (2006)
11. Holt, J.K., Chung, S.M.: Mining association rules using inverted hashing and pruning. Information Processing Letters 83, 211–220 (2002)
12. Halkidi, M.: On clustering validation techniques. Journal of Information Systems 17, 107–145 (2001)
13. Hearst, M.A.: Untangling text data mining. In: Proceeding ACL 1999 Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics on Computational Linguistics, pp. 3–10 (1999)

14. Massey, L.: Evaluating quality of text clustering with ART1. In: Proceedings of the International Joint Conference on Neural Networks, vol. 2, pp. 20–24 (2003)
15. Steinbach, M., Karypis, G., Kumar, V.: A comparison of document clustering technique. In: Grobelnik, M. (ed.) KDD Workshop on Text Mining, Boston (2002), `http://www.cs.cmu.edu/~dunja/KDDpapers/Steinbach-IR.pdf`
16. Hastie, T., Tibshirani, R.: The elements of statistical learning: data mining, inference and prediction. The Mathematical Intelligencer 27(2), 83–85 (2008), doi:10.1007/BF02985802
17. Wu, B., Zheng, Y., Liu, S.H.: CSIM: a document clustering algorithm based on swarm intelligence evolutionary computation. In: 2002 World Congress on Computational Intelligence, Honolulu, pp. 477–482 (2002)
18. Tsay, Y.-J., Chang-Chien, Y.-W.: An efficient cluster and decomposition algorithm for mining association rules. Information Science 160, 161–171 (2004)