FAQ Finder for Computer Related Questions

Lwin Mar Oo, Nang Saing Moon Kham
University of Computer Studies, Yangon
lwinmar.lwinmaroo@gmail.com *;*

Abstract

This paper presents Frequently Asked Question (FAQ) finder system that uses a natural language question based on interface to access text information sources specifically text files organized as question-answer pairs such as FAQ files. FAQ finder is a natural language question answering system that uses files of frequently asked questions as its knowledge base. It identifies the files that are relevant to the query and then matches against the segments of text that are used to organize the files themselves. The similarity algorithms is to create as a search engine in this system, where all this information is gathered and used to find the best matching and method for a specific request. And its main purpose is for information retrieval systems to get the most relevant results. It uses existing resources of "questions asked" and "answers given" to provide a simple and natural interface between users and information networks. It will implement a FAQ finder system that will provide immediate natural language access to a large corpus of computer and accessories information.

1. Introduction

In the vast information space of the internet, groups have created small amount of questions, around their particular interests and hobbies. Most FAQs are periodically posted on the newsgroups to which they are relevant. This information distribution mechanism works well for individuals who are sufficiently interested in a topic to subscribe to its newsgroup.

Question answering (QA) systems can be seen as information retrieval systems which aim at responding to natural language queries by returning an appropriate answers rather than lists of documents. The idea behind a FAQ file is to record the consensus of opinion among a group on some common question and make that answer available, particularly to newcomers to the group who may otherwise ask the same questions again and again.

The aim of the FAQ Finder is to construct a question-answering system that extends further the intent of the FAQ file phenomenon. The system is an information service, available on the World-Wide Web, to which users can pose questions. FAQ Finder answers a user question by searching FAQ files for a similar question. If a successful match is found between a user question and a FAQ question, then the answer supplied in the FAQ file is returned to the user.

This paper is organized as follows. Section 1 is the introduction, section 2 is related work. Section 3 is the proposed system design and the components of the system and theories used in this system. Section 4 is the system implementation and case study for FAQ finder system. Section 5 describes the conclusions of the system.

2. Related Work

FAQ Finder compares the question to its set of FAQ files, returning a list of files ranked by their relevance to the question. Unlike Artificial Intelligence question answering systems that focus on generation of new answers, FAQ answering systems retrieve existing question-answer pairs from their databases. FAQ Finder [1] is a system designed in order to improve navigation through already existing external FAQ collections. The system has an index – FAQ text files organized into questions, section headings, keywords, etc. In order to match a user question to the FAQs, the system (1) does syntactic parsing of the question, identifies verb and noun phrases in the question, and (2) performs semantic concept matching in order to select possible matches between the question and target FAQs in the index.

The famous FAQ retrieval systems include FAQ Finder [3], Auto-FAQ [5], Sneiders' system [4], and Ask Jeeves[6]. In FAQ Finder system, vector-space model (VSM) is used to calculate similarity and WordNet is used to perform concept matching. In Auto-FAQ system, techniques in natural language processing are adopted to improve the performance of keyword comparison. In Sneiders' system, to match users' queries to FAQ collection, keywords are classified into required keywords, optional keywords, and irrelevant keywords. In Ask Jeeves system, the FAQ collection is classified into 11 classes. Then keywords of user queries are used to search for relevant FAQs. In some researches, case-based reasoning (CBR) method is adopted to find a set of rules, and user queries will be added to FAQ collection incrementally to propose a dynamic retrieval method [2]. But, it can not process uncertain classification efficiently.

3. Proposed System

This paper presents FAQ finder system that uses FAQ files to fulfill the user natural language query. In this system, a user will enter a question in natural language and the system will attempt to find an information source that answers the question, and then find the closest matching question/answer pair. These systems combine three technologies, statistically based Information Retrieval (IR) engines syntactic language analysis and semantic networks. SMART (System for the Mechanical Analysis and Retrieval of Text) information retrieval system is based on Cosine Similarity Algorithm, a natural language parser is based on the tree tagger and a semantic net is derived from WordNet, are combined for the FAQ Finder System. Overview system design is shown in Figure 1.

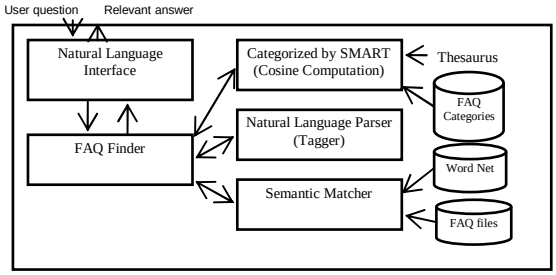

Figure 1: Overview of the System Design

3.1 Components of the System

This system combines three approaches for answering the user question.

- First user's query (question) is categorized by SMART IR system based on cosine similarity algorithm, which is used to perform the initial step of narrowing the focus to a small subset of the FAQ files.
- After categorizing, user's query is parsed into Natural language parser. It will tag each word in the question in the form of tree structure.
- Finally, relevant question is searched by query matching algorithm. In this system, Semantic matcher will be used. WordNet dataset will be used to find synset ,that is defined as a set of one or more synonyms (such as **computer**, computing machine, computing device, data processor, electronic computer, information processing system -- (a machine for performing calculations automatically)), for the semantic values.

Process flow of the proposed system is shown in Figure 2.

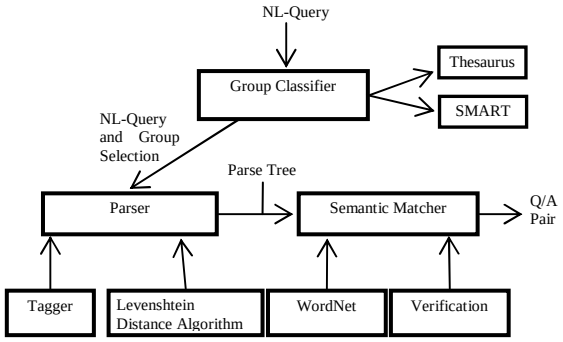

Figure 2: Process Flow of the System

3.2 SMART Information Retrieval based on Cosine Similarity Algorithm

SMART IR System is a classification algorithm using Cosine Similarity algorithm. It is a vector based approach and weights are calculated from Term Frequency (TF) and Inverse Document Frequency (IDF) value of each term in the user question. The user's question is treated as a query to be matched against the library of FAQ files. SMART stems all the words in the query and removes those on its stop list of frequent words. It then forms a term vector from the query, which is matched against similar vectors already created for the FAQ files in an offline indexing step. The top-ranked files from this procedure are returned to the user for selection.

It is fairly accurate at locating set of relevant files. Question templates are used to assign a question class to each question type. FAQ finder tries to find questions in the files that belong to the same class and contain the same terms. The system can look for descriptive questions for each of the terms separately. General model of Smart is shown in Figure 3.

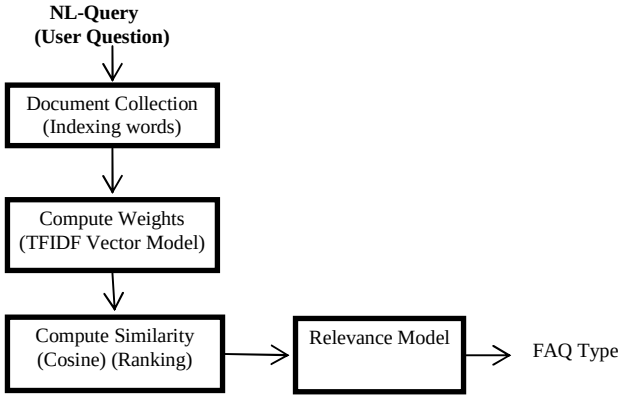

Figure 3: General Model of SMART

The process flow of the SMART is that, the question is defined by the user, is extracted as meaningful words and removed the common words. That is the **indexing** step. Then the system performs

the **weighting** step that is each term is assigned weight value by TF/ IDF.

The last step is **ranking** that is based on similarity measure. In this step, the cosine equation and product of the document and query are included.

Cosine Similarity Algorithm

Cosine similarity is used in the matching process of SMART IR System. It is a similarity algorithm to compute the similarity between user query and question categories. It is a vector space model and TF and IDF values of user question are matched against the indexes of question categories. Similarity value is computed by following equations.

$$\text{Sim}(a, b) = \cos\theta = \frac{a \bullet b}{|a| \times |b|}$$

$$\text{Sim}(S_a, S_b) = \frac{\sum_{i=1}^{n}(S_{ai} \bullet S_{bi})}{\sqrt{\sum_{i=1}^{n} S^2_{ai} * \sum_{i=1}^{n} S^2_{bi}}}$$

Whereas a · b entitles the dot product and is calculated by multiplying term weights of the query- and document vector together. The second equation is the similarity that is a series being summed, where each term is the product of the weight in the document vector times the weight in the query vector for the given keyword divided by the square root of the product of the series being summed of the square of the weight in the document vector and the series being summed of the square of the weight in the query vector.

The result value of that similarity equation is the possible to be from 0 to 1. The similarity value nearly 1 is the most relevant category of the system.

3.3 Natural Language Tagger and Parser

In this proposed system, tree tagger tool is used to tag the user question in the natural language tags. It is a part of speech tagger, which is the processing of the question to choose the correct tags by the syntactical roles. Tagged text is input for the parser. Resulting parsed tree is used in two ways:

- To categorize the question according to a set of question type
- To assign generic semantic cases to noun phrases

The own grammar is designed and include the question-type as non-terminal symbols. Example tree tagger is shown in Figure 4.

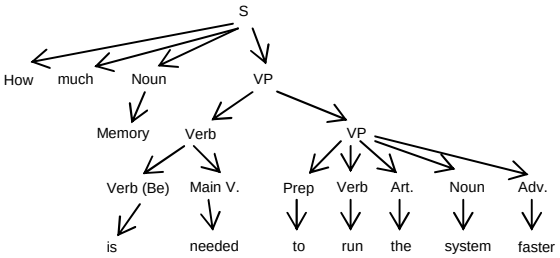

Figure 4: Parsing into Tagger

Levenshtein Distance Algorithm

The Levenshtein distance is the difference between two strings. Levenshtein distance (LD) is a measure of the similarity between two strings, which can be referred to as the source string (s) and the target string (t). The distance is the number of deletions, insertions, or substitutions required to transform s into t. The greater the Levenshtein distance, the more different the strings are. The implementation of computing distance is shown in Figure 5.

```
double ComputeEditDistance(string s, string t)
  int n = s.Length;
  int m = t.Length;
  int[,] distance = new int[n + 1, m + 1];
  int cost = 0;

  if (n == 0) return m;
  if (m == 0) return n;

  for (int i = 0; i <= n; distance[i, 0] = i++) ;
  for (int j = 0; j <= m; distance[0, j] = j++) ;


  for (int i = 1; i <= n; i++)
 {
   for (int j = 1; j <= m; j++)
   {
    cost = (t.Substring(j - 1, 1) == s.Substring(i - 1, 1) ? 0 : 1);
// all cost op of 1
        distance[i, j] = Min3(distance[i - 1, j] + 1,
        distance[i, j - 1] + 1,
        distance[i - 1, j - 1] + cost);
    }
   }

   return distance[n, m];
```

Figure 5: Distance Computation Algorithm

3.4 Matching Algorithm

In this system, matching algorithm (Cosine Similarity) is used to find an appropriate answer from the FAQ files. Then WordNet database is used to compute shallow lexical semantics in this system. WordNet provides much of the lexical information found in dictionaries of English in a form that can be used by computers for processing natural language (NLP). WordNet entries consist of a set of synonyms

(a synset). WordNet 2.1 database is used in this system.

4. System Implementation

This system is implemented using Microsoft Visual Studio 2008 with ASP .NET C#. It is a web-based FAQ system. User can post new FAQ entries and search existing FAQ entry. This paper mainly focuses on finding relevant FAQ. Existing FAQs are categorized by their category (computer maintenance, computer virus, etc.) and question type (comparison, how to, etc.). Terms and their TF values are stored in the indexing database to compute the similarity values. IDF values are computed only when user posts a query. FAQs used in this system are collected from http://www.answers.com[7]. There are 100 FAQs for each category and there are total 600 FAQs in the knowledge base. Database structure design is shown in Figure 6.

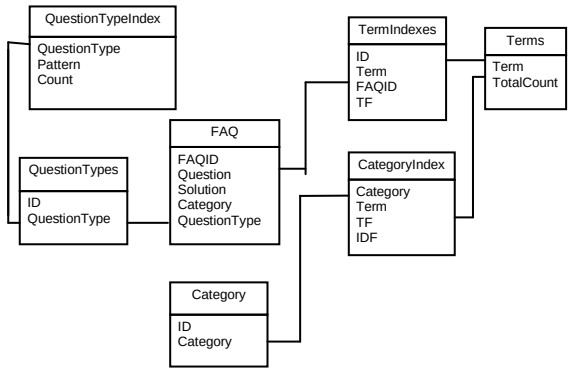

Figure 6: Database Design of the Proposed System

4.1. Case Study

In the knowledge base has about one hundred questions in each question category and each question type. Some categories and FAQ questions are used in this case study. Its knowledge base is created by the following steps. Firstly, Table 1 shows Categories of the proposed system and Table 2 is Question Types data of that system.

Table 1: Categories

ID	Category
1	Other
2	Computer Hardware
3	Computer Maintenance
4	Computer Networking
5	Computer Programming
6	Computer Viruses
7	Internet

Table 2: Question Types

ID	QuestionType
1	Other
2	Comparisons
3	Normal Questions
4	How To
5	Reason

The data sets of this system are defined by the system developer. The FAQ files for this system are shown as Table 3. In these FAQ files, the developer assigns the question and answer pairs. The question and answer pairs are defined as their category and question type by the developer.

Table 3: FAQ Files

ID	Question	Solution	Category	QuestionType
1	Why does a computer run slow?	Because of presence of...	Computer Maintenance	Reason
2	How do you reformat windows?	This guide will wipe ..	Computer Maintenance	How To
3	What are the advantages and disadvantages of computers?	The Internet can help you do ...	Computer Maintenance	Comparisons
4	How do you get rid of Trojan Spooner A virus?	Download and run firefox to...	Computer Viruses	How To
5	What is a modem?	A modem is a Modulator …	Computer Hardware	Normal Questions
6	How do you remove the virus called iexplore.exe - Bad Image?	Your system has been seriously...	Computer Viruses	How To

Then the system calculates that question's TF. If the user's question is entered in that system, the system will removed the common words and extracted the meaningful words by SMART. It is shown by the following table.

Table 4: Term Indexes for Category

Category	Term	TF
Computer Hardware	computer	1
Computer Hardware	modem	1
Computer Hardware	port	1
Computer Hardware	power	1
Computer Hardware	problem	1
Computer Hardware	serial	1
Computer Hardware	supply	1
Computer Maintenance	defragment	1
Computer Maintenance	disk	1
Computer Maintenance	freeze	1

Computer Maintenance	format	1
Computer Maintenance	reboot	1
Computer Maintenance	windows	1
Computer Viruses	Trojan	1
Computer Viruses	Virus	1
Computer Viruses	exe	1

For Example: The user enters the question "How do you solve Trojan virus?" The system indexes the extracted words and finds the weight value of the query and document as Table 5. Next step, the system calculates the cosine similarity. Then the result of the question category can be got.

User Question: How do you solve Trojan virus?

The user question is categorized by cosine similarity algorithm. There are three categories which are computer hardware, computer maintenance and computer virus. The system removes the common words and extracts the meaningful keywords 'Trojan' and 'Virus'. That is the **indexing** step in SMART IR system.

No. of categories that 'Trojan' contains = 1

No. of categories that 'Virus' contains = 1

Then weight values (TF x IDF) of terms 'Trojan' and 'Virus' are described in Table 5.That step is defined the **weighting** step in SMART.

Table 5: The Weight Value of the Query and Document of the User Question

	Trojan (TF x IDF)	Virus (TF x IDF)
Question	1	1
Computer Hardware	0	0
Computer Maintenance	0	0
Computer Virus	1 x log(3 / 1) = 0.477	1x log (3 / 1) = 0.477

After computing the weight values of the terms, that system compute the similarity of these terms by using the cosine similarity algorithm. The final step is the **ranking** step in SMART.

Sim (Q, Computer Hardware) = 0

Sim (Q, Computer Maintenance) = 0

Sim (Q, Computer Virus) = (0.477 + 0.477) / Sqrt (2 x 0.455)

= 0.954 / Sqrt (0.91)

= 0.954 / 0.954 = 1

The system can be resulted as "User Question belongs to **Computer Virus**." These above steps are how to create the knowledge base in this system by using SMART IR system with indexing, weighting and ranking steps.

If the category is resulted, the tree tagger tagged the input sentence and then parsed by the syntactic parser. Then the tagger tool recorded the question templates from the questions defined by the system developer. Then the Levenshtein distance algorithm gives the result of the distance value of the question type as in Table 6. Threshold value is assigned to set the question type. If the least distance value is more than threshold value, the question type is not possible to be assigned.

The following expression is the question templates of the user question.

<ex> SUBJ VB PP </ex>

Table 6: Computing Edit Distance of Question Type

Question Type	Distance Value
Comparisons	0.7500
Normal Questions	0.7500
How To	0.2400
Reason	0.4300

According to distance values in Table 6, user question belongs to question type "**How To**".

4.2. Experimental Results

The quality of prioritized keyword matching algorithm is determined by its ability to retrieve relevant existing FAQs from the database upon a user's request. For Information Retrieval process, there are two parameters to measure the quality of such retrieval – recall and precision. Recall R characterizes the system's ability to retrieve all the relevant items existing in the collection of documents (i.e., FAQs):

$$R = \frac{\text{numbers of relevant documents retrieved}}{\text{total numbers of relevant documents in collection}}$$

Precision P characterizes the system's ability to retrieve only relevant items:

$$P = \frac{\text{number of relevant documents retrieved}}{\text{total number of documents retrieved}}$$

For our system, there are several possible reasons for why this is the case. First, the SMART can get round about 81.54% recall and 78% precision for the similarity because TFIDF value will be changed depending on the input questions in data sets. The parser found a parse for 80.32% recall and 77.25% precision of the user questions; it is possible that parser sometimes found the wrong parse or mis-tags words. Matching algorithm alone gets 75.87% recall and 73.45% precision and this algorithm alone degrades the processing time since it does not filter questions, i.e., it processes all FAQ in the knowledge base. And then the output of the whole system is

about 87.12% recall and 86.23% precision because this system uses combination of above three technologies, which makes improving overall accuracy.

5. Conclusion

FAQ finder rely on the knowledge engineering inherent in FAQ files distributed on the different locations. System can show the existing collection of questions and answers found in the FAQ files to the user. The aim is to make the answer recorded in FAQ files more widely available. The users can get the information, what the user should know, from this system when the user wants to know about the related computers and their accessories and how to work them and, how to maintain them.

6. References

[1] Burke R., Hammond K., Kulyukin V., Lytinen S., Tomuro N. and Schoenberg S., "Question Answering from Frequently Asked Question Files: Experiences with the FAQ Finder System", AI Magazine, vol. 18, no. 2: 57-66, 1997.

[2] Fu Y. and Shen R., "GA based CBR approach in Q&A system", Expert Systems with Applications, vol. 26, Issue 2. pp.167-170, 2004.

[3] Hammond K. , Burke R., Martin C., and Lytinen S., "FAQ Finder: A case-based approach to knowledge navigation". In The Proceedings of the 1995 IEEE Conference on Artificial Intelligence Applications. IEEE, IEEE Press, February 1995.

[4] Sneiders, E., "Automated FAQ answering: continued experience with shallow language understanding", AAAI Fall Symposium, pp.97–107, 1999.

[5] Whitehead S. D. , "Auto-FAQ: An experiment in Cyberspace leveraging", Computer Networks and ISDN Systems, vol. 28(1–2), pp.137–146, 1995.

[6] Ask Jeeves, http://www.ask.com

[7] FAQ files are collected from http://www.answers.com