FAQchat as an Information Retrieval System

Bayan Abu Shawar, Eric Atwell and Andrew Roberts

School of Computing
University of Leeds, LS2 9JT, UK
{bshawar, eric, andyr}@comp.leeds.ac.uk

Abstract

A chatbot is a conversational agent that interacts with users through natural languages. In this paper, we describe a new way to access information using a chatbot. The FAQ in the School of Computing at the University of Leeds has been used to retrain the ALICE chatbot system, producing FAQchat. The results returned from FAQchat are similar to ones generated by search engines such as Google. For evaluation, a comparison was made between FAQchat and Google. The main objective is to demonstrate that FAQchat is a viable alternative to Google and it can be used as a tool to access FAQ databases.

1. Introduction

Human computer interfaces are created to facilitate communication between human and computers in a user friendly way. For instances information retrieval systems such as Google are used to remotely access and search a large information system based on keyword matching. However, the best interface is arguably one which fools you into thinking that you are speaking/asking a real human; a chatbot.

A chatbot is a conversational software agent, which interacts with users using natural language. The idea of chatbot systems originated in the Massachusetts Institute of Technology, where Weizenbaum implemented the ELIZA chatbot to emulate a psychotherapist (Weizenbaum, 1966). After ELIZA, a lot of chatbots or human-computer dialogue systems have been developed either to simulate different fictional or real personalities such as PARRY (Colby, 1999) to simulate a paranoid patient, or to be used as an interface to help systems or web-based search engines such as AskJevees (2004). We have worked with the ALICE open-source chatbot initiative. ALICE (ALICE, 2002; Wallace, 2003) is the Artificial Linguistic Internet Computer Entity, developed by Wallace in 1995. In the ALICE architecture, the "chatbot engine" and the "language knowledge model" are clearly separated, so that alternative language knowledge models can be plugged and played. We have techniques for developing new language models, to chat around a specific topic: the techniques involve machine learning from a training corpus of dialogue transcripts, so the resulting chatbot chats in the style of the training corpus.

User input is effectively used to search the training corpus for a nearest match, and the corresponding reply is output. We adapted this chatbot-training program to the FAQ in the School of Computing (SoC) at University of Leeds, producing the FAQchat system. The results returned from FAQchat are similar to ones generated by search engines such as Google, where the outcomes are links to exact or nearest match web pages. A search engine is "a program that searches documents for specific keywords and returns a list of the documents where the keywords were found." (Internet.com, 2004). However FAQchat could also give direct answers and the algorithm underlying each tool is different.

Section 2 describes ALICE architecture. Section 3 presents a brief introduction about the previous work. To evaluate FAQchat, a comparison was made between the FAQchat and Google presented in section 4. Section 5 describes the methodology of evaluation. Results are discussed in sections 6. Section 7 presents our conclusion that FAQchat is a viable alternative to Google in accessing FAQ databases.

2. ALICE system architecture

ALICE stores knowledge about English conversation patterns in AIML files. AIML, or Artificial Intelligence Mark-up Language, is a derivative of Extensible Mark-up Language (XML). It was developed by the Alicebot free software community during 1995-2000 to enable people to input dialogue pattern knowledge into chatbots based on the ALICE free software technology. AIML consists of data objects called AIML objects, which are made up of units called topics and categories. The topic is an optional top-level element, it has a name attribute and a set of categories related to that topic. Categories are the basic unit of knowledge in AIML. Each category is a rule for matching an input and converting to an output, and consists of a pattern, which represents the user input, and a template, which implies the ALICE robot answer. The AIML pattern is simple, consisting only of words, spaces, and the wildcard symbols _ and *. The words may consist of letters and numerals, but no other characters. Words are separated by a single space, and the wildcard characters function like words. The pattern language is case invariant. The idea of the pattern matching technique is based on finding the best, longest, pattern match.

2.1 Types of ALICE/AIML categories

There are three types of categories: atomic categories, default categories, and recursive categories.

Atomic categories are those with patterns that do not have wildcard symbols, _ and *, e.g.:

```
<category><pattern>10 Dollars</pattern>
<template>Wow, that is cheap!</template>
</category>
```

In the above category, if the user inputs '10 dollars', then ALICE answers 'Wow, that is cheap'.

Default categories are those with patterns having wildcard symbols * or _. The wildcard symbols match any input but they differ in their alphabetical order. Assuming the previous input 10 Dollars, if the robot does

not find the previous category with an atomic pattern, then it will try to find a category with a default pattern such as:

<category><pattern>10 *</pattern>
<template>It is ten.</template> </category>

So ALICE answers 'It is ten'.

Recursive categories are those with templates having <srai> and <sr> tags, which refer to simply recursive artificial intelligence, and symbolic reduction. Recursive categories have many applications: symbolic reduction that reduces complex grammatical forms to simpler ones; divide and conquer that splits an input into two or more subparts, and combines the responses to each; and dealing with synonyms by mapping different ways of saying the same thing to the same reply as the following example:

<category><pattern>HIYA</pattern>
<template><srai>Hello</srai></template>
</category>

The input is mapped to another form, which has the same meaning.

2.2 ALICE/AIML pattern matching technique

The AIML interpreter tries to match word by word to obtain the longest pattern match, as this is normally the best one. This behaviour can be described in terms of the Graphmaster set of files and directories, which has a set of nodes called nodemappers and branches representing the first words of all patterns and wildcard symbols. Assume the user input starts with word X and the root of this tree structure is a folder of the file system that contains all patterns and templates; the pattern matching algorithm uses depth first search techniques:

If the folder has a subfolder starting with underscore then turn to, "_/", scan through it to match all words suffixed X, if no match then:

Go back to folder, try to find a subfolder starts with word X, if so turn to "X/", scan for matching the tail of X, if no match then:

Go back to the folder, try to find a subfolder start with star notation, if so, turn to "*/", try all remaining suffixes of input following "X" to see if one match. If no match was found, change directory back to the parent of this folder, and put "X" back on the head of the input. When a match is found, the process stops, and the template that belongs to that category is processed by the interpreter to construct the output.

3. Previous work

A Java program was developed to converts the readable text (corpus) to the chatbot language model format. Two versions of the program were generated. The first version is based on simple pattern template category, so the first turn of the speech is the pattern to be matched with the user input, and the second is the template that holds the robot answer. This version was tested using the English-language Dialogue Diversity Corpus (DDC) (Mann, 2002), to investigate the problems of utilising dialogue corpora (Abu Shawar and Atwell, 2003a). The learning techniques range from primitive literal matches to corpus utterances, to more complicated patterns involving identification of the most significant words in an utterance (Abu Shawar and Atwell, 2003b).

In the first word approach we assumed that the first word of an utterance may be a good clue to an appropriate response: if we cannot match the input against a complete corpus utterance, then at least we can try matching just the first word of a corpus utterance. For each atomic pattern, we generated a default version that holds the first word followed by wildcard to match any text, and then associated it with the same atomic template.

The first word approach was tested using the Corpus of Spoken Afrikaans (Rooy, 2002). Unfortunately this approach still failed to satisfy our trial users, so we looked for the word in the utterance with the highest "information content", the word that is most specific to this utterance compared to other utterances in the corpus. This should be the word that has the lowest frequency in the rest of the corpus. The most significant approach was selected to generate the default categories, because usually in human dialogues the intent of the speakers is hiding in the least-frequent, highest-information word. The program calculates the Afrikaans corpus word-frequency list, and then a comparison is run against each token in each pattern to find the least frequent word with that pattern. Four categories holding the most significant word were added to handle the positions of this word first, middle, last or alone. The feedback showed improvement in user satisfaction (Abu Shawar and Atwell, 2003c).

4. Comparing the FAQchat with Google

Google is "a search engine which is very easy to use. It returns pages based on the number of sites linking to them and how often they are visited, indicating their popularity." (SeniorNet, 2004). Search engines like Google retrieve information in four phases (Boyle, 2003):

1. Obtaining documents to be searched. The method used gives a classification of search engine types:
   a. Search engines which use crawlers, or spiders to get URLs such as Google;
   b. Search engines based on human submission;
   c. Others that are a combination of the two.
2. Preparing the documents to be searched, which involve operations such as: filtering the text, and extracting the meaningful items.
3. Indexing the items. One of the mechanisms used by Google is the inverted file structure. Three stages are applied here:
   a. Each document has a unique ID;
   b. A dictionary of all stemmed words from all documents is created.
   c. Each item in the dictionary is associated with a pointer to the inversion list. The inversion list associates each item to all documents containing it.
4. The matching process to give the best answer to a specific user query. One of the most widely used methods is the vector space model, where a two-dimensional array (term by document) is constructed with size M x N; M represents the items in dictionary, and N represents the documents. A weighting scheme may be applied such as column normalisation or tf-idf. The users query is

represented as a vector of size M, and it is normalised, stemmed, and weighted in the same manner as the document's items. At the end the best hit will be selected using different methods of ranking. This ranking algorithm is the "hidden ingredient" differentiating rival search engines.

Most search engines break up the user query into keywords, and return results according to keyword matches like Google. Ask Jeeves (2004) is a search engine that returns a result after understanding the query, using a question-processing engine to understand the meaning of the words and grammar of the question. FAQchat is a compromise between the two. In retrieving information FAQchat will try to give the results using most significant words as keywords, and try to find the longest pattern to match without using any linguistic tools, or analysing the meaning. FAQchat does not need a linguistic knowledge module, and also in principle is language independent: it can be trained with FAQs in any natural language. The way FAQchat works is described below:

1. All questions and answers are extracted from the whole database after applying a filtering process to remove unnecessary tags.
2. The FAQ database yielded questions and (answers). A list of links is constructed, containing the links from FAQ to web pages containing answers.
3. A dictionary is created, containing all words in the questions with frequencies of occurrence. Then the first and second most significant words are extracted from each question.
4. AIML pattern-matching rules, known as "categories", are created. There are two possible types of match: input matches a complete FAQ question; or input matches 1st or 2nd most significant word in an FAQ question (least frequent words). There are two types of responses generated, which either has the direct answer (in the instance where only one match was found) or if the most significant words are found in more than one question, multiple links are returned as a reply.

The aim of this evaluation is to show that FAQchat works properly; it is not a search engine, but it could be a tool to access web pages, and giving answers from FAQ databases. The aim is not specifically to measure comparative success of Google against FAQchat, but merely to demonstrate the FAQchat is a viable alternative. Moreover, the most significant word approach has already been used to develop earlier versions of the chatbot, which deal with text and dialogues. The aim of this experiment is to show that the same approach is applicable with the FAQ database.

5. Evaluation methodology

To evaluate FAQchat, an interface was built, which has a box to accept the user input, and a button to send this to the system. The outcomes appear in two columns: one holds the FAQchat answers, and the other is holds the Google answers after filtering it to the FAQ database. Google allows search to be restricted to a given URL, but this still yields all matches from the whole SoC website (http://www.comp.leeds.ac.uk) so a Perl script was required to exclude matches not from the FAQ sub-pages.

An evaluation sheet was prepared which contains 15 information-seeking tasks or questions on a range of different topics related to the FAQ database. The tasks were suggested by a range of users including SoC staff and research students to cover the three possibilities where the FAQchat could find a direct answer, links to more than one possible answer, and where the FAQchat could not find any answer. In order not to restrict users to these tasks, and not to be biased to specific topics, the evaluation sheet included spaces for users to try 5 additional tasks or questions of their own choosing. Users were free to decide exactly what input-string to give to FAQchat to find an answer: they were not required to type questions verbatim; users were free to try more than once: if no appropriate answer was found; users could reformulate the query.

The evaluation sheet was distributed among 21 members of the staff and students. Users were asked to try using the system, and state whether they were able to find answers using the FAQchat responses, or using the Google responses; and which of the two they preferred and why.

6. Results

Twenty-one users tried the system; nine members of the staff and the rest were postgraduates. The analysis was tackled in two directions: the preference and the number of matches found per question and per user.

6.1 Number of matches per question

The number of evaluators who managed to find answers by FAQchat and Google was counted, for each question.

Results in table 1 shows that 68% overall of our sample of users managed to find answers using the FAQchat while 46% found it by Google. Since there is no specific format to ask the question, there are cases where some users could find answers while others could not. The success in finding answers is based on the way the questions were presented to FAQchat.

Users /Tool	Mean of users finding answers		Proportion of finding answers	
	FAQchat	Google	FAQchat	Google
Staff	5.53	3.87	61%	43%
Student	8.8	5.87	73%	49%
Overall	14.3	9.73	68%	46%

Table 1: Proportion of users finding answers

Of the overall sample, the staff outcome shows that 61% were able to find answers by FAQchat where 73% of students managed to do so; students were more successful than staff.

6.2 The preferred tool per each question

For each question, users were asked to state which tool they preferred to use to find the answer. The proportion of users who preferred each tool was calculated. Results in figure 1 shows that 51% of the

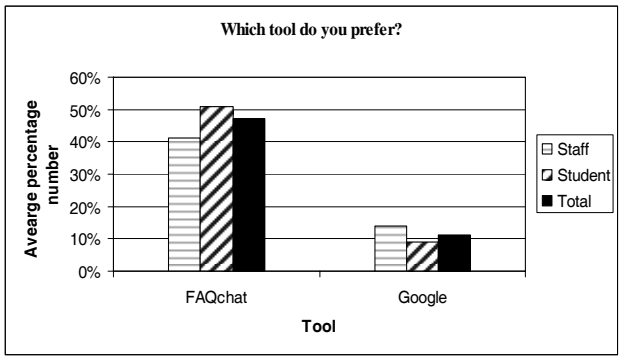

Figure 1: Proportion of preferred tool

staff, 41% of the students, and 47% overall preferred using FAQchat against 11% who preferred the Google.

6.3 Number of matches and preference found per user

The number of answers each user had found was counted. The proportions found were the same. The evaluation sheet ended with an open section inviting general feedback. The following is a summary of the feedback we obtained:

1. Both staff and students preferred using the FAQchat for two main reasons:
   a. The ability to give direct answers sometimes while Google only gives links.
   b. The number of links returned by the FAQchat is less than those returned by Google for some questions, which saves time browsing/searching.
2. Users who preferred Google justified their preference for two reasons:
   a. Prior familiarity with using Google.
   b. FAQchat seemed harder to steer with carefully chosen keywords, but more often did well on the first try. This happens because FAQchat gives answers if the keyword matches a significant word. The same will occur if you reformulate the question and the FAQchat matches the same word. However Google may give different answers in this case.

6.4 Samples of chatting

In this section samples of asking questions to FAQchat and Google are presented which show the different answer possibilities: direct answer, or a set of related links as shown in figure 2 and figure 3.

---

a. Direct answer generated by FAQchat using the FAQ of the school of computing at University of Leeds

Human: How can I remove a job from the printer?
FAQchat:
If you want to remove a job from the printer queue then find out the job number using the lpq command, and in a UNIX shell type:

% **lprm -P** *printername job_no*

You can only remove jobs that you have sent to the printer yourself - apart from Support that is.

For more information look at:
How do I remove a job from the printer queue? (Printing)

b. Related links generated by FAQchat

Human: How can I delete a job from a printer?
FAQchat: You may find your answer in the following links:
How do I check the printer queue? (Printing)
How do I remove a job from the printer queue? (Printing)
How do I check my printer credits? (Printing
How do I use the printer from a Windows machine? (Other Windows information)

Figure 2: Answers generated by FAQchat

FAQchat could generate either a direct answer or related links for the same question. This is because the matching process is based on matching the most significant words found in the user input. In figure 2 sample a, the significant words are "remove", and "printer" in addition to the first word "how". Where in sample b, the significant word is "printer" only in addition to the first word "how". So different answers could be generated based on the way the question is formulated.

Figure 3 shows that both Google's and FAQchat's answer is the same as the top answer in produce by FAQchat. However, FAQchat provides all links related to courseworks, where link number three holds the answer for the user's input.

---

a. Answers generated by FAQchat

Human: What happens if i can't submit the coursework on time?
FAQchat: You may find your answer in the following links:
Examinations/Coursework (General Information)

Why coursework? (Examinations/Coursework)

What if I can't hand coursework in on time? (Examinations/Coursework)

Can students be granted extensions or exemptions from coursework? (Teaching)

What are the penalties for late submission of coursework? (Teaching)

b. Answers generated by Google for the same question

Examinations/**Coursework**
... 5. What happens **if** I **can't** attend my exam **...** DOESN'T **HAPPEN**!! **... If** a student is considered by the School Examinations Board to have made no serious attempt to pass a **...**
www.comp.leeds.ac.uk/faq/public/x0704.html - 14k - Supplemental Result - Cached - Similar pages

---

Figure 3: Answers generated by FAQchat and Google

7. Conclusions

Feedback favourable to FAQchat was gained from almost all users, even those who preferred Google. Overall, they found it a novel and interesting way to access the FAQ using natural language questions. Overall, about two thirds of users managed to find answers by FAQchat, and about two thirds of the users preferred to use it.

The aim was not to try to evaluate the two systems to come up with relative scores, but to show that it is a viable alternative way of Google and it could be used as a tool to access FAQ databases.

8. References

ALICE. (2002). A.L.I.C.E AI Foundation, http://www.Alicebot.org/

Abu Shawar, Bayan; Atwell, Eric. (2003a). Using dialogue corpora to train a chatbot. In: Archer, D, Rayson, P, Wilson, A & McEnery, T (editors) *Proceedings of Corpus Linguistics 2003*, pp. 681-690

Abu Shawar, Bayan; Atwell, Eric. (2003b). Machine learning from dialogue corpora to generate chatbots. *Expert Update*, vol. 6, pp. 25-30.

Abu Shawar, Bayan; Atwell, Eric. (2003c). Using the corpus of Spoken Afrikaans to generate an Afrikaans chatbot. *Southern African Linguistics and Applied Language Studies*. Vol. 21, pp. 283-294.

Abu Shawar, Bayan; Atwell, Eric. (2004). An Arabic chatbot giving answers from the Qur'an. In: Bel, B and Marlien, I (eds.) *Proceedings of TALN04*. Vol 2, pp. 197-202 ATALA.

AskJevees. (2004). [Online]: http://ask.co.uk/home

Boyle, Roger. (2003). Understanding search engines. In: Boyle, R (ed) *COMP1600: SY11 Introduction to Computer Systems 1*. Lecture Notes. School of Computing, University of Leeds, UK. pp 65-72.

Colby, K. (1999). Human-computer conversation in a cognitive therapy program. In: Wilks, Y. (ed.) *Machine conversations*. Kluwer, London. pp 9-19.

Internet.com. (2004). Search engine. [Online]: http://www.webopedia.com/TERM/s/search_engine.htm

Mann, W. (2002). Dialog Diversity Corpus.
http://wwwrcf.usc.edu/~billmann/diversity/DDivers-site.htm

SeniorNet. (2004). Lesson 4. [Online]: http://www.seniornet.org/php/default.php?ClassOrgID=5337&PageID=5920

Van Rooy, B. (2002). *Transkripsiehandleiding van die Korpus Gesproke Afrikaans*. [Transcription Manual of the Corpus Spoken Afrikaans.] Potchefstroom: Potchefstroom University

Wallace, R. (2003) The elements of AIML style. ALICE AI Foundation.

Weizenbaum, J. (1966). ELIZA – A computer program for the study of natural language communication between man and machine. *Communications of the ACM*. Vol. 10, No. 8, pp 36-45.

Martin, L.E. (1990). Knowledge Extraction. In Proceedings of the Twelfth Annual Conference of the Cognitive Science Society (pp. 252--262). Hillsdale, NJ: Lawrence Erlbaum Associates.