Conversation System with State Information

Elyse Marie Glina, Byeong-Ho Kang
emglina@utas.edu.au, bhkang@utas.edu.au
University of Tasmania

Abstract

Most current approaches to conversation system development invoke a complex set of language parsing rules or development tools difficult for novices to handle and are unable to convincingly simulate advanced natural language features such as topic awareness or conversation thread involvement. This study proposes an alternate approach based on the ripple down rules (RDR) algorithm, presently used to enable more effective maintenance of expert systems. This tree-based algorithm enables a conversation system to travel incrementally deeper into a particular topic, then to switch based on context-dependent information to the correct previously discussed topic – resulting in a highly reusable method of developing conversation systems based around a variety of topics.

1. Introduction

Efforts over the decades to enable humans and computers to communicate and collaborate through natural language have taken two main approaches: large-scale scientifically or academically focused efforts attempting to handle all potential inputs, and projects with more limited focus and a particular objective, usually commercial and advertising-related.

The latter of these is of particular interest to this study. Presently, while there are systems available that facilitate the development of low-level systems, none of these are particularly friendly to inexperienced would-be developers, or enable a developer of any skill level to easily see the capabilities and weaknesses of the conversation system as it develops. This limits the concept's value to amateur or low-budget projects.

This study demonstrates that the ripple down rules (RDR) algorithm can potentially be adapted to produce a base that can be used to rapidly develop and deploy a variety of styles of conversation system. Although not conventionally used in this manner, RDR is suitable for such adaptation as it is founded on the idea that all statements are contextual. Likewise, the truth values of statements that occur in natural language are highly dependent on preceding statements. The resulting approach is far more consolidated than other available approaches and therefore appropriate for development of low-level conversation systems in terms of effort required. It addresses many of the issues faced by conversation systems from this different angle, while the approach also explores problem areas not solved by current conversation systems.

However, this study also demonstrates that a "pure" RDR implementation is not sufficient to solve all the unique problems faced by a conversation system, though it can achieve a significant level of effectiveness against effort compared to other systems as it is easily supplemented with reusable modules for exceptional situations. In addition, the nature of

context in an interaction and the search tree traversal process that should be applied may depend on the intended objective of the conversation system being built.

2. Background

2.1.Expert Systems

An expert system can apply the reasoning and inference techniques first implemented by knowledge-based systems, used in exploration of the knowledge encompassed by the problem – as opposed to clear-cut algorithmic solutions[1)] – in order to emulate the problem-solving functions of a human expert in the problem field the system has been developed to address.[2)]

Expert systems generally consist of several interrelated parts. Of particular importance are the knowledge base, which comprises the knowledge set of the expert, and the inference engine, which is analogous to the 'brain' of the system, performing operations on the expert knowledge in the knowledge base. Different classes of individuals interact with the system: the domain expert, who is the human counterpart of the expert system; the knowledge engineer who translates domain expert knowledge into a form the expert system can work with; and, potentially, outside users who have neither domain experience or technical experience, who will work with the expert system in order to acquire useful solutions to problems in its domain.[3)]

The main difficulty faced in developing expert systems is negotiation between the entities required to maintain the system. Different experts think differently about how to solve the same problem, creating difficulties when experts collaborate on a single system.[4)] The need for the expert to communicate their poorly-understood reasoning processes to the knowledge engineer in order to improve the effectiveness of the expert system results in a heavy limit on how much new knowledge can be added to the system per day, termed the 'bottleneck problem'.[2)] Experts are also prone to overspecialisation or overgeneralisation when making classifications, which may result in a system that does not meet their needs. Conventional expert systems struggle to support rectification of such problems.[5)]

2.2. RDR

The RDR algorithm was developed to solve the above difficulties with expert systems.

RDR operates on the idea that a fact's truth value is only defined by the original context it is provided in, and therefore should only ever be evaluated in that context. An RDR expert system's knowledge base is therefore built as a tree structure, whose individual points or nodes represent a particular classification. The link between the parent and child nodes is the rule that must fire to produce that classification.[4)] The knowledge base begins empty and is built up while the system is in use. If a classification the system gives at a particular point is not specific enough for the expert's purposes – or is even incorrect – then a refinement rule is linked to the relevant rule, only evaluated in that rule's context, which will provide a more satisfactory classification. Rules are never deleted or modified once added to the database, only refined in this manner until the classification process of the system is acceptable. A certain level of redundancy is therefore expected in a Ripple Down Rules system, acknowledging the multiple ways in which any expert problem may be solved.[6)] Such a system thus supports multiple experts' input, simultaneously or at different times.

Most importantly, the straightforward nature of the RDR expert system enables experts to perform their own operations on the system without requiring a dedicated knowledge engineer. This has generated proven results in real-world implementations.[7)]

2.3. Conversation Systems

Most current conversation systems operate by defining common related sentence structures and keywords, and linking these to the appropriate responses that may be made when faced with input of this style. AIML, the XML-based development backend for well-known current conversation system A.L.I.C.E., from which several other successful online conversation systems have been developed, uses these methods.[8)]

The common weakness of most current and past conversation systems up to now is their lack of awareness of the topic they are discussing and their poor performance when switching between topics. This generally reveals that they merely respond to the statements provided by the human participant.[9)] Systems that do implement topic awareness do not treat this as a primary objective or impose heavy structure over the process.[8)]

3. Method

3.1. Objective

The desired outcome of developing a conversation system is project-dependent, making the success of any system difficult to evaluate outside of a real-life scenario, as it may depend on a combination of factors.

Considering these factors, the focus of this study was not to develop a conversation system able to compete with current top-tier systems such as A.L.I.C.E.. Instead the objective was to examine the characteristics that have made RDR-based expert systems successful over others: namely their convenience for domain experts lacking computing expertise and the ability to modify the produced system's granularity as required, The results of applying these characteristics to the task of conversation system construction were explored by building a basic framework for conversation system development, which was then used to construct a series of conversation systems feasibly representing how such a system might be used.

3.2. Developed System

The system developed consisted of two main parts: the database and inference engine. An additional element, designed to deal with exceptional circumstances, was developed but eventually determined to be beyond the scope of the study except for illustrative purposes.

3.2.1. Database

The database is in a human-readable format with each entry containing four fields: the unique identifier, the unique identifier of its "parent" entry, a "trigger" text and a response text. Identifier data enables the tree structure to be reconstructed as required. The trigger text is analogous to the classification requirements in a traditional RDR system. As such, it may consist of an entire natural language sentence, but in an attempt to catch multiple alternate phrasings of the same statement that occur frequently in imprecise natural language, most of the trigger text entries in the developed system are words, phrases or lists of words and phrases – separated by delimiters and to be evaluated separately – which should always generate the same response in the system. The response text is the exact statement the system responds with. It can be seen that the only operation that will be performed on user-entered text is basic pattern-matching, and that responses are not generated dynamically. The result of this is that the grammatically garbled responses that are a frequently observed result of stringent testing of a conversation system[9)] will not occur. Implementing sufficient appearance of "language diversity" in such a system could be of future concern, but this is not a primary objective of this study.

3.2.2. Inference Engine

The inference engine is the most significant indication of how the Ripple Down Rules-based conversation system differs to other systems. The engine's basic operation is to construct a search tree on-the-fly from the parts of the active database that are invoked, and to then attempt to search incrementally one node deeper into this tree per interaction between user and system. At each step the last node returned as a response – equivalent to the current point in the conversation – and all the nodes visited in the current conversation are recorded.

The engine searches the children of the current node for a match with the successive user entry. If no match is found at any point during the conversation, the engine backs through the tree, searching any unexplored children of explored nodes for a match. In the context of a natural language conversation, this represents a topic change. If the match is near the root of the tree this may be a new topic. If the match is some distance down an explored path it is likely to be a topic previously discussed at some length. The correct division of the conversation database by context is expected to enable questions without specific context such as, "How does it work?" or, "Tell me more", to be processed correctly, giving the system the appearance of a sense of conversation flow, while enabling user entries which provide further context to be handled separately.

If no match is found, then the system will record this information for later viewing by an expert. On the expert side, the inference engine enables rapid addition of new rules to the database.

3.2.3. Exceptions

The final element of the system, which was implemented for illustrative purposes but not involved in testing, is handling of 'exceptional' situations. Emotional responses such as frustration and confusion carry connotations that, in a natural language interaction between two humans, would result in a break from basic processing and a jump to a handling process that would put the conversation back on track. Similarly, situations such as a request to repeat a previously requested answer require a different type of keyword operation. The developed

system demonstrates how keywords indicating these states can be searched for before the operation of the main inference engine and alternate operations on the tree invoked.

4. Results and Discussion

4.1. Database Building

The initial test conducted on the developed system was to attempt to construct a very simple conversation system from a previously established table of required information, based on a domain of well-known characteristics that might be used in a commercial setting.

Immediately apparent from this initial building up of the conversation system database was that, although the idea of context is intuitively simple, determining exactly what information should be considered "in context" in the early stages of building up a database is not. The system may be subjected to two conceivable types of questions. "Hostile" questions will provide a large amount of contextual keywords or phrases in a single statement, or else assume some context is already obvious from the nature of the system, thus challenging the system's ability to process to the correct depth. "Friendly" questions may offer only one keyword at a time, but are equally challenging to handle as they require all possible levels of context to be explicitly handled in the database in order for the correct response to be available.

As a consequence of needing to handle these extremes of interactions and also all questions that might fall in between, the initial construction of the database involved a high level of repetition of similar rules. Although a certain amount of repetition is considered an advantage in RDR expert systems, the expectation is that this repetition will occur on isolated occasions, and will not require more-than-doubling of the amount of rules entered initially to enable the system to function usefully.

4.2. Evaluation in Operation

Meaningful evaluation of the success of the developed system in practice was difficult, as such evaluation should rightfully take place in a real-world situation over a period of time outside of the scope of this study. The developed conversation system achieved reasonable success when developed with the anticipation of "hostile" questions and when subsequently questioned in a "hostile" manner. A weakness identified during this testing was the expectation that keywords comprising key phrases would be sequential, a decision based on the requirement of the system to be simple enough for an uninitiated user to learn quickly.

Accounting for the "friendly" question style proved to be the biggest difficulty for the system. Handling all possible "hostile" questions during initial database development is simple, but after taking this step the handling of "friendly" questions quickly expands the database to a level of complexity making it difficult to track obvious missing strings of context except through live experimentation. This is undesirable as live experimentation for the RDR conversation system has different connotations than it does to the expert system – the conversation system is designed to be exposed to the public. At the levels of redundancy necessary to cover a reasonable amount of "friendly" questions in even a very small conversation system, it becomes questionable whether anything of significance is being demonstrated, as the potential and thus far unproven benefit of a topic-aware conversation system is then unquestionably outweighed by the effort required to produce it.

5. Conclusions

Although conversation systems developed with the RDR base were clearly inferior when comparing the effort that would be required to build a similar conversation system with an alternative building tool such as AIML, the very different principles of the approach raised useful questions about the future of conversation system research, Some of the system's failures can be attributed to the lack of focus and previous data in this initial study, rather than a general failure of the approach.

With its clear success in deploying systems suited to asking of direct questions, the system appears well-suited to deployment in situations where its users have already been coached to ask direct questions only. However, using it in this way would not enable it to take advantage of any of the potential benefits previously laid claim to, and would probably render the development environment excessive for the resulting systems developed. A perceived objective of a RDR-based conversation system is to break free of the current truism that conversation systems that work within a particular domain (or topic) are more effective.[9)] Conversation systems covering a particular topic are, however, likely to be highly desirable in the commercial settings for which such rapid conversation system development seems tailored, and the system can still demonstrate its main strengths in this area by allowing very fine-grained control of subtopics within a domain. In this context it would be considered acceptable to tailor systems to a certain extent towards the objectives they intend to achieve.

Implementing the inference engine's main process to attempt to advance a single node deeper into the tree per user/system interaction was chosen merely for simplicity. It would, for example, be acceptable to develop a system that could travel through multiple levels of the tree per interaction. Such a system might be able to resolve the issue of excessive redundancy by enabling a single fully enumerated context span to handle both "friendly" and "hostile" lines of questioning.

What is remarkable about the RDR approach to conversation system development is that it produces neither a conversation system nor an expert system relating to the development of conversation systems, but rather a system that has the basic capabilities to teach its users how to develop a conversation system, by virtue of its simple structuring. The success of the approach in other domains, and the simplicity with which its philosophy is mapped to the philosophy that can be inferred from natural language operation, warrants further exploration of the method.

6. Further Work

The potential of the RDR framework to guide a user through the process of developing a tailored conversation system is presently untapped, due largely to lack of available data on this new process. A necessary step to ensure relative consistency of success in development of RDR-based systems would be to gather data on such factors as the level of redundancy necessary in the system, the most effective levels of tree depth and the appropriate deployment of trigger text, and incorporate the knowledge gained into the system. In this way it could potentially offer guidance instead of merely showing the user the overall structure of what has been developed and trusting them to notice mistakes that may not be apparent to a non-technical user.

If exploration of more guided conversation system development should prove fruitful, then it would be worthwhile to implement a more complex pattern matching process in such a system, allowing for words to be excluded or matched even if not sequential. The assumption for this study was that providing such an array of options would distract from the purpose of providing a simple development environment. However, a more involved version of the system might be able to provide enough assistance to counter this effect.

7. References

1) R. Davis, "Knowledge Based Systems", Science, American Association for the Advancement of Science, New York, 231-4741, 957-963, (1986).

2) E. Feigenbaum, A. Bond (ed.) "Expert systems in the 1980s", State of the art report on machine intelligence, Pergamon Infotech, New York, (1981).

3) D. Merritt, Building Expert Systems In Prolog, Springer-Verlag, (1989).

4) P. Compton, R. Jansen, "A philosophical basis for knowledge acquisition", *Knowledge Acquisition*, 2-3, 241-258, (1990).

5) T. Cao, P. Compton, "A simulation framework for knowledge acquisition evaluation", P. of the Twenty-Eighth Australasian Conference on Computer Science, 38, 353-336, (2005).

6) P. Compton, L. Peters, et al, "Experience with Ripple-Down-Rules", *Knowledge-Based Systems*, 19, 356-362, (2006).

7) G. Edwards, P. Compton, et al, "Piers: A pathologist-maintained expert system for the interpretation of chemical pathology reports", Pathology, 25-1, 27-34, (1993).

8) A.L.I.C.E. AI Foundation, "A.L.I.C.E.", http://www.alicebot.org (last accessed 2009).

9) D. Coniam, "Evaluating the language resources of chatbots for their potential in English as a second language", ReCALL, 20-1, 98-116, (2008).