View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by OAR@UM

Building a Hybrid: Chatterbot - Dialog System

Alexiei Dingli and Darren Scerri

Department of Intelligent Computer Systems, University of Malta
{alexiei.dingli,dsce0006}@um.edu.mt

Abstract. Generic conversational agents often use hard-coded stimulus-response data to generate responses, for which little to no effort is attributed to effectively understand and comprehend the input. The limitation of these types of systems is obvious: the general and linguistic knowledge of the system is limited to what the developer of the system explicitly defined. Therefore, a system which analyses user input at a deeper level of abstraction which backs its knowledge with common sense information will essentially result in a system that is capable of providing more adequate responses which in turn result in a better overall user experience.

From this premise, a framework was proposed, and a working prototype was implemented upon this framework. The prototype makes use of various natural language processing tools, online and offline knowledge bases, and other information sources, to enable it to comprehend and construct relevant responses.

Keywords: Dialog Systems, External Knowledge Acquisition, RDF, Knowledge Bases, Chatterbots

1 Introduction and Background

Conversational agents are deployed in various forms and designed to cater for different domains and goals, ranging from automated hotel booking agents, to personal assistants, companions and entertainment purposes. Moreover, one can categorise conversational agents into two main types based on how these process the input and generate their output. These two types can be realised as being *chatterbots* and *dialog systems* [5].

The main difference between these two types of conversational agents lies in what these systems are designed to model. Chatterbots model, or rather simulate a conversation in its basic sense, and intend to fool the user that he is communicating with an intelligent entity that does in fact understand what is being said. On the other hand, dialog systems attempt to model the actual dialog process which also incorporates the task of analyzing and understanding the input, which in turn aids in the generation of an adequate dynamic response.

Due to the nature of conversational agents, and by implication, the notion of natural language, the user is given the opportunity to provide less restricted input compared to other types of systems using other more conventional types of user interfaces. Therefore, conversational agents must provide a certain tolerance with regards to unexpected input, in other words, *robustness*. This issue is normally tackled by the use of stimulus-response methodology, or rather, pattern-matching techniques in chatterbots, since this provides a certain level of control over the system for which other approaches are not currently able to provide. Dialog systems carry out more complex input analysis to achieve a more meaningful representation of the input than simple pattern-matching techniques [5].

Conversational agents used mainly for generic conversation, *chatterbots*, depend on large amounts of inflexible language data while dialog systems make use of more refined technologies and approaches, including the integration of knowledge and the use of methods originating from Computational Linguistics [5]. It has also been shown that dialog systems which make use of certain knowledge bases and ontologies benefit in terms of *recognition*, *interpretation*, and *generation* [8]. Even though this study has been carried out on strict domains, the same approach could be used for broader and generic domains, similar to those which are typically associated with chatterbots.

Studies have shown that dialog systems which make use of an open world model produce a "more realistic conversation than a system without the open world model" [2]. This notion refers to the ability of a dialog system to use external knowledge bases, and in certain cases unstructured text, to gather knowledge about topics, concepts and entities which it does not have any information about. By the presence of this feature, dialog systems are able to more deeply understand the input provided by the user, and by implication, respond in a more intelligent and dynamic way, unlike simpler stimulus-response agents, which respond in a pre-defined verbatim manner.

2 Aims and Objectives

> "Building a system that could understand open-ended natural language utterances would require **common sense reasoning**, the huge open-ended mass of sensory-motor competencies, **knowledge and reasoning skills** which human beings make use of in their everyday dealings with the world" [6].

The aim of this project is to provide a proof-of-concept generic-conversation (unrestricted domain) framework for conversational agents, and a working prototype which can be categorised as being a hybrid between a chatterbot and a dialog system.

With regards to this hypothesis, the system makes use of modern natural language processing technologies and tools to analyze user dialog input while simultaneously using information that is obtainable from external sources to attempt to further under-stand the input and ultimately generate appropriate responses to the user. Another goal is to merge these various sources to create a single, local, knowledge base which enables the system to keep track of the world of the user, i.e. the relationship and interaction of the user with various entities. These sources include content and information on different entities and concepts, including common sense knowledge and knowledge about specific people and world entities.

The aims and objectives of this project can be summarized as follows:

1. To build an expandable proof-of-concept system that provides a syntactic, semantic and pragmatic understanding of input.
2. To simulate intelligence by providing adequate output and logical conclusions de-rived from dialogue input and local and external knowledge bases.

Moreover, the proposed system will be as customizable, flexible, and modular as possible, so that it would require minimum effort to upgrade and adapt the system to handle input of varying complexity and topics.

3 Design

The developed prototype consists mainly of three phases: Natural Language Understanding (NLU) phase, intermediary processing phase, and finally the output generation phase.

The NLU module is primarily makes use of ChatScript [9], an award-winning chatterbot engine. ChatScript employs various linguistic technologies to enable support for more flexible linguistic input in terms of syntax and semantics, and more expressive and semantics-oriented pattern matching rules, where matching patterns of meaning (semantics and pragmatics) is considered more important than matching patterns of words (syntax). These technologies and processes include the use of WordNet as a semantic network, Part-of-Speech tagging, pronoun resolution, conceptual relations, and preprocessing abilities.

ChatScript is used by the system mainly as a “normalisation phase”. This phase is essentially a mapping process that maps natural language input into a more formal representation using XML. This is essential in order to allow the creation of rules that allow matching of a number of input utterances that are effectively semantically equivalent. This phase is analogous to how Faade maps text to discourse acts.

Input is preprocessed using two approaches. Using GATE and ANNIE [3, 1], the system attempts to resolve pronouns into their respective named entities. Moreover, ChatScript includes a preprocessing phase in itself, performing actions such as spell checking and term substitution.

The normalisation phase allows the system to perform further intermediary processing on the input, such as querying local and external knowledge bases to allow better input comprehension. The conversational agent forwards the input to ChatScript after some surface pre-processing for which in turn it expects a normalised input in order to allow the system to further process the input. This

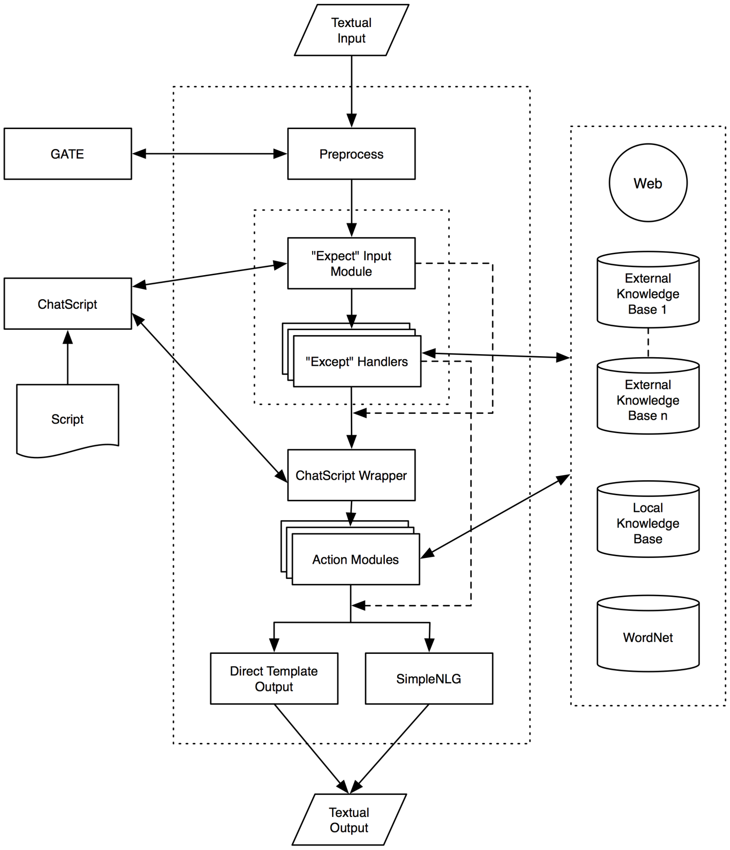

Fig. 1. System Block Diagram

is achieved by a number of processing modules (called "Action Modules") that can be developed to handle specific types of input.

For example, an Action Module can be designed and developed to handle any form of *"who"* questions. Upon an utterance which can be viewed as being a *"who"* question is detected by ChatScript, the input is then normalised and forwarded to the conversational agent for processing. The agent will then appoint an action module (in this case, the *"who" Action Module* to process that input and ultimately generate a response which can be dynamically generated from external knowledge bases (Figure 2).

The system's dialog manager employs a similar technique used in CONVERSE [4] with regards to these processing modules. The system incorporates a polling technique for which the input is passed through all implemented processing modules, or Action Modules, for which each module advertises a score indicating how much it is confident to process the input.

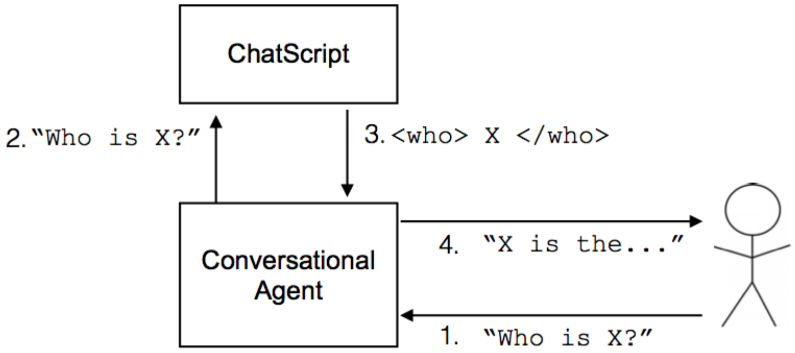

Fig. 2. Data flow between conversational agent and ChatScript

Each action module is able to perform various tasks, include querying and managing local and external knowledge bases through global wrappers, triggering other action modules, querying ChatScript for further normalisation, adding and retrieving topics from the dialog manager, generating output and alerting the dialog manager that the subsequent user utterance is expected to contain certain type of information (such as names, feelings, numbers, etc.).

The knowledge bases used in the system can be grouped into two types: the local RDF knowledge base, and other external knowledge bases (can be either offline or online), which are all accessible throughout the system. The local RDF knowledge base is managed using Apache Jena [7] and backed by its TDB component.

YAGO and ConceptNet are used by the system as external knowledge bases allowing the system to gather and make use of both entity specific and common sense knowledge. The retrieved data from these knowledge bases can be merged with the local RDF knowledge base to create an amalgamated network of knowledge, combining the world of the user with external knowledge.

4 Implementation and Evaluation

A prototype system was developed upon the described framework. This prototype included the development of a number of action modules, in conjunction with their respective ChatScript script which attempt to handle a specific set of user utterances, ranging from possession and sentiment statements, to questions regarding general and specific world entities.

The evaluation was carried out two-fold: hands-on conversation with the system (prototype), and a questionnaire filled by a number of users who engaged in a dialogue with the system, following a pre-defined context and a set of example tasks, tested this prototype and answered questions on a five-point Likert scale

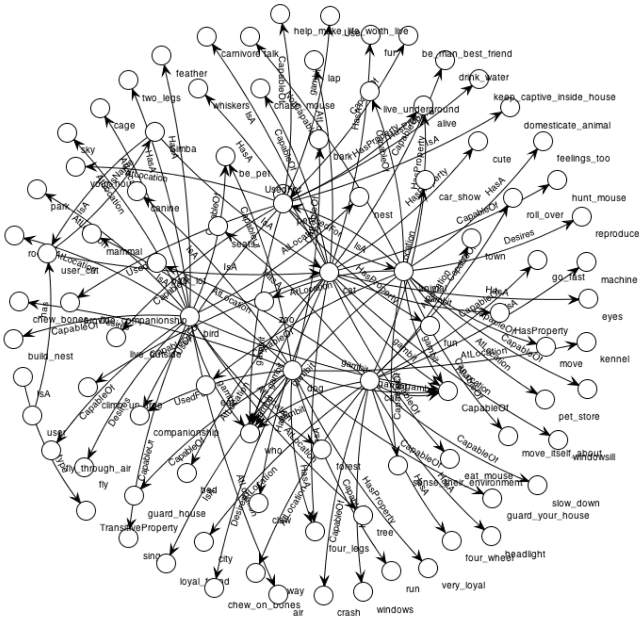

Fig. 3. An Example of a Generated RDF Graph

which were aimed to be conclusive in whether the system actually achieved its aims and objectives. All results were biased towards the higher end (mean values for answers: 1 - 2.05%, 2 - 6.77%, 3 - 25.84%, 4 - 49.73%, 5 - 15.54%), which imply that the overall result is positive.

The following example shows the system's ability to keep track of user-defined in-formation and depicts a level of understandability:

User: I have a red car.
System: OK, noted!
User: What is the color of my car?
System: Red.
User: I also have a cat named Simba.
System: Glad to know that you have a cat named Simba.
User: Do you know who is Simba?
System: Simba is your cat.
User: Do you know who is Barack Obama?
System: Barack Hussein Obama II is the 44th and current President of the United States. Also, I have heard a lot of positive things about Barack Obama. What's your view?
User: I think he is a great president .
System: Why do you think that Barack Obama is great?
User: Because he is an exemplary politician.
System: Ok then, so you think that Barack Obama is great, because he is an exemplary politician.
User: Are cats capable of chasing mice?

```
System: Yes
User: Can cats fly?
System: No, I don't think so
User: Is a cat an animal?
System: Yes
User: Is it a machine?
System: No
User: Do you know anything about Bill Gates?
System: Bill Gates is born on date 1955−10−28.
User: Anything else?
System: Bill Gates is married to Melinda Gates.
User: Tell me about cats
System: Cats are capable of eating meats, killing birds, drinking milks, eating
    mice, sleeping, climbing up trees and chasing mice.
```

5 Conclusions and Future Work

It is concluded that the results obtained are positive and show that the proposed framework has potential in supporting the implementation of dynamic conversational agents where generic knowledge need not be explicitly defined in the system beforehand as hard-coded data. Satisfying the project's aims and objectives for open-ended conversation would require a system which surpasses the ability to pass the Turing Test since it would also require responding in an informed and knowledgeable manner, and not just fooling a human that the system is itself a human and not a machine.

Naturally, there are still possible areas for improvement since the evaluation was done on a prototype with strict boundaries. Possible future work can be carried out in the integration of more refined dialog management techniques which will enable the system to be used in practical environments especially in complicated task-based environments. Another area for which the system falls short is its NLG capabilities.

Moreover, information extraction techniques can be used to gather information about generic and specific entities for which data is not found in the structured external knowledge bases.

An agent similar to Siri, i.e. a personal assistant, can be developed using this framework and the implemented modules, since such agent would not require complex dialog management techniques, but simply answering to the user's requests, in a task-oriented environment. If such system is to be developed, one would need to adapt the system to not only be able to reply, but also is able to talk to the user when necessary, e.g. as a reminder.

In addition, the system can be further improved by the implementation of new modules to achieve higher quality results in terms of input comprehension, mainly with regards to semantics and therefore, the actual meaning of the user utterances.

The system's ability to make use of external sources to support and enhance its knowledge of the real world can be considered as being a step forward towards the implementation of more natural and human-like conversational systems. The system is able to exploit the vast amounts of data found in structured knowledge databases that are consulted to both understand, and answer in a natural and informed manner.

References

1. K. Bontcheva, H. Cunningham, D. Maynard, V. Tablan, and H. Saggion. Developing reusable and robust language processing components for information systems using gate. In *In 3rd International Workshop on Natural Language and Information Systems (NLIS'2002). Aix-en-Provence*, pages 223–227. Society Press, 2002.
2. R. Catizone and Y. Wilks. A companionable agent. *Conversational Agents and Natural Language Interaction Techniques and Effective Practices*, page 302, 2011.
3. H. Cunningham, K. Humphreys, and R. Gaizauskas. Gate - a tipster-based general architecture for text engineering. In *In Proceedings of the TIPSTER Text Program (Phase III) 6 Month Workshop. DARPA*. Morgan Kaufmann, 1997.
4. B. L. C. David Levy, B. Batacharia, R. Catizone, A. Krotov, and Y. Wilks. Converse: a conversational companion. In *in Proc. of the 1st International Workshop on Human-Computer Conversation*, 1997.
5. T. Klüwer. From chatbots to dialog systems. *International Journal*, pages 1–22, 2011.
6. M. Mateas and A. Stern. Natural language understanding in façade: Surface-text processing. 2004.
7. B. McBride. Jena: A semantic web toolkit. *IEEE Internet Computing*, 6(6):55–59, Nov. 2002.
8. D. Milward and et al. Ontology-based dialogue systems, 2003.
9. B. Wilcox. Chatscript documentation. `http://chatscript.sourceforge.net/Documentation/`, 2012.