“Eliza, a friend indeed” Project Report

Kevin Higgins

General Problem

Creating a Conversationalist bot which can interact with people over an IM client (in this case Pidgin). The bot will be proficient in the domain of Pokemon primarily by searching the Internet. The ultimate goal is to have the bot pass the Turing test.

Specific Solution

In order to allow my existing Eliza bot to interface with a Pidgin client, I decided to find a Perl module to assist me. I found Net::OSCAR created by Matthew Sachs on CPAN. To utilize the module, I simply create an OSCAR object, tell it what function to call when a text message is sent (im_in), and log in to the chosen AIM account.

As a learning experience, I turned my Eliza bot into a Perl module called Bot::Giles. To use it, simply declare a Giles object, and send a message using talk_giles.

To use both modules together, I make a call to talk_giles within the im_in function. Thus, whenever OSCAR receives a text message, it sends it to giles, who returns his message. im_in then sends the message back to whoever sent the initial message.

Whenever Giles receives a message from a user, he checks the user name against a hash of users he has spoken to in the past. When Giles starts up in fact, names from past sessions are loaded from a file to be used in conversation. Currently, this is primarily to recognize the fact that Giles knows the user. If Giles does not know a user, he greets them and asks them for their name. This is then added to his database of buddies.

If a known user messages Giles, the users state is evaluated, to see if there is anything pertinent for Giles to say. For example, if Giles just met someone, and learned their name, he will continue with an introduction to the world of Pokemon. This feature can be expanded to allow for any event to trigger a state to be called upon the next user query.

The next phase of query evaluation is to determine if the query involves a specific Pokemon. If so, the Pokeget package will identify the type of answer, and tell Giles. These specifically include the Pokemon's physiology, habitat, what it evolves into, what evolves into it, it's type (element sorta), color, Pokemon number, or a random it of trivia about it.

If no Pokemon info is to be shared, Giles checks to see if the user is referencing him. If so, he will randomly present the user with information regarding himself.

Next, the module parse_search.pm allow Giles to access the Internet if a question is asked regarding something that isn't a specific Pokemon. Making a call to get_web_info($sentence) from parse_search.pm will find the nouns and verbs of a query using regular expressions and Lingua::EN::VerbTense. These are appended to a string. This string is then cross referenced by web pages found via LWP get requests to Google. Originally, Yahoo API performed this duty, but it stopped working. The first sentence found by this cross reference is then returned to Giles to be sent to the user.

Finally, Giles performs some basic word matching and manipulation a la Eliza if he can't come up with anything to say. If the query doesn't even satisfy these requirements, he will resort to his delightful barrel of phrases he keeps at the ready.

Related Work

Artificial Paranoia – Kenneth Mark Colby

Summary:
This paper discusses a bot name Parry created to simulate the response of a paranoid hospital patient. Most interesting to me about this bot was the system used to determine the ideal response to a query. The system uses artificial measurements of anger, fear and mistrust to respond. This is unique, because the bot must have the ability to judge different inputs in order to raise levels of anger/fear, consequently raising the level of mistrust. Through this model, the bot is able to give responses similar to that of a paranoid human, since it has “emotions” similar to that of a paranoid individual being questioned.

Relation:
This idea of a bot program having certain states seems to be key to creating a believable AI. User input affecting bot output past the initial response is key to holding up a conversation. Although a human can be fooled for a little while simply by being responded to. It eventually becomes apparent that the correspondent lacks memory, and consequently humanity. I'm under the impression, that this was a significant gap in my program. I made my focus on simply responding to the most recent query, and mistakenly ignored the conversation as a whole. Although I did implement a state system for my program, I did not use it to make as much of an impact as could have been.

ELIZA – Joseph Weizenbaum

Summary:
This paper discusses the inner workings of the bot “ELIZA”. This is obviously notable since it was one of the first attempts at passing the Turing test. It simulates a Rogerian therapist, using NLP tricks like word matching and returning reformed input back at the user. It is also interesting to note that the paper concludes by mentioning how improvements can be made to the old ELIZA. It suggests making use of a store if information to allow Eliza to better form responses. Additionally, she should use a belief structure to learn input from a large spectrum of users, helping to form a knowledge base, not preset within the program.

Relation:
The Eliza paper had a significant impact on my bot, since it's implementation was founded on my original attempt at an Eliza bot. Word matching and query mutation was used for this, in addition to

some canned default responses. I really like the idea of learning a belief structure from a user base so as to better facilitate future responses. It would be interesting as applied to my project, but truly, with an Internet connection, it's almost better to just find a reliable website to find facts. Perhaps if my bot was meant have a more broad topic range it would be very interesting though. One topic is possible to cover most bases for, but beyond that, it's far less work to teach the bought.

Scaling Question Answering to the Web - Cody Kwoc

Summary:
Mulder is truly an impressive undertaking. Using several phases, the system is able to zero in on the answer to a query. It starts off by parsing a query. It then classifies the query, and makes a query reformulation wherein the question is mutated to be more like the potential answer to the question. After parallel submission to a search engine, answer extraction from the resulting documents occurs by comparing parts of the text to the expected answer type. The potential answers are then rated, and the best one is selected.

Relation:
I made an attempt at question reformulation in my own project, combined with the use of a search engine, and my own form of answer extraction. Answer selection was simply first come first used. My question reformulation did conjugate verbs, and use nouns to help create keywords for a query, which was kind of along the same lines as Mulder. Sadly though, it wasn't terribly effective, although mildly entertaining. Additionally, my answer extraction was simply a hunt for a sentence containing the same key words as the reformulated query. I'd have loved to implement a more advanced form of the algorithm, but did not allow myself the opportunity.

AI Chat Bots Developing – Ehab El-agizy, Moustafa Zamzam

Summary:
This paper seemed to be more broad then the previous works. It talked about bot emotion states, similar to the Colby paper, answering questions via belief structuring. It also went into some depth on other areas of potential bot AI uses. Not necessarily just passing the Turing test, but having the bot interact with a computer via system commands (scary) and performing tasks based on user input.

Relation:
I found this paper to be good as perhaps an introduction to bot programming. It doesn't elaborate very much on any of the topics it covers, but it provides a nice window into the world of bot programming, and allows someone new to the field to find out what particular area they'd like to focus on, and continue their research elsewhere. Particularly, I thought the papers discussion on state was thought provoking, and intend to use it in combination with the Colby paper to extend the state functionality I do have in the future.