Information Extraction Tools and Methods for Understanding Dialogue in a Companion

R. Catizone, A. Dingli, H. Pinto, Y. Wilks

Computer Science Department

University of Sheffield

E-mail: roberta@dcs.shef.ac.uk, alexiei@dingli.org, h.pinto@dcs.shef.ac.uk, y.wilks@dcs.sef.ac.uk

ABSTRACT

This paper discusses how Information Extraction is used to understand and manage Dialogue in the EU-funded Companions project. This will be discussed with respect to the Senior Companion, one of two applications under development in the EU-funded Companions project. Over the last few years, research in human-computer dialogue systems has increased and much attention has focused on applying learning methods to improving a key part of any dialogue system, namely the dialogue manager. Since the dialogue manager in all dialogue systems relies heavily on the quality of the semantic interpretation of the user's utterance, our research in the Companions project, focuses on how to improve the semantic interpretation and combine it with knowledge from the Knowledge Base to increase the performance of the Dialogue Manager. Traditionally the semantic interpretation of a user utterance is handled by a natural language understanding module which embodies a variety of natural language processing techniques, from sentence splitting, to full parsing. In this paper we discuss the use of a variety of NLU processes and in particular Information Extraction as a key part of the NLU module in order to improve performance of the dialogue manager and hence the overall dialogue system.

1. Introduction

Over the last few years, research in human-computer dialogue systems has increased and much attention has focused on applying learning methods to improving a key part of any dialogue system, namely the dialogue manager (Young, S., 2006). Since the dialogue manager in all dialogue systems relies heavily on the quality of the semantic interpretation of the user's utterance, our research in the Companions project, focuses on how to improve the semantic interpretation and combine it with knowledge from the Knowledge Base to increase the performance of the Dialogue Manager. Traditionally the semantic interpretation of a user utterance is handled by a natural language understanding module which embodies a variety of natural language processing techniques, from sentence splitting, to full parsing. In this paper we discuss the use of a variety of NLU processes and in particular Information Extraction as a key part of the NLU module in order to improve performance of the dialogue manager and hence the overall dialogue system.

2. The Senior Companion

The Senior Companion (Figures 1a, 1b) is one of two applications under development in the EU-funded Framework 6 Companions Project. It is designed to facilitate senior citizens in carrying out everyday tasks and providing easy access to information, including past conversations. It is intended to be deployed in a variety of devices, from computer desktops to handheld devices and small robots. The current implementation of the Senior Companion is one of two applications under development and provides a means for engaging in multimodal human-computer dialogue by discussing personal photographs. The scenario is first and foremost a means for building a life narrative of a person through the act of reminiscing; albeit restricted to people, places, events and memories recorded through and triggered by a set of photos. The user interacts with the system via voice or text and has the added feature of being able to read current news (Figure 1c) (taken from online news feeds) to the user (though the system does not converse about the news). The system runs on a touchscreen laptop and so embodies the modality of touch as well as text/speech.

The construction of an enjoyable and cohesive interaction between the user and the computer in the Senior Companion sets the scene for advanced multimodal dialogue research and in particular the research discussed in this paper.

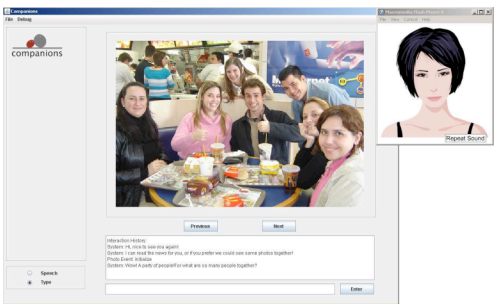

Figure 1a: Senior Companion screenshot version 1

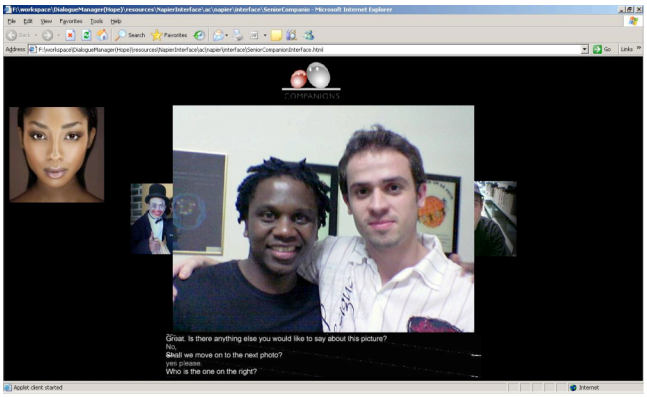

Figure 1b: Senior Companion screenshot version 2

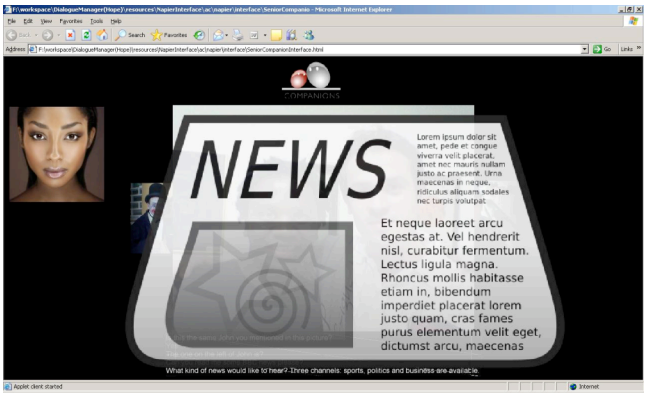

Figure 1c: Senior Companion news reading screenshot

The Senior Companion is a multimodal dialogue system for discussing information about the users life in a natural way through photographs. Although we are aiming to discuss a broader range of topics later in the project, to begin we have identified a basic selection of topics that will allow the system to engage in a lively dialogue and establish a platform for reasoning about user information. The initial topics we have selected for engaging in a dialogue about a user's photographs are: the location of photograph, when the photo was taken and for what occasion and the names and ages of the people in the photos as well as their relationship to the user and to each other. Not surprisingly, the key information for each of the topics mentioned includes Named Entities: Person Names, Locations, Dates/Times, People_Relationships. etc. Based on this, we have chosen an Information Extraction approach to recognize and extract such information from the user's utterance which is explained in more detail in the following sections of this paper. Figure 2 contains a very simple user session taken from an early version of the demonstrator.

S: Hello, what is your name?
U: My name is Roberta.
S: Thank you Roberta. Let's begin to look at your pictures now. Who is this person?
U: It's John.
S: How old is John?
U: He was 20 in this picture I think.
S: How do you know John?
U: He is my best friend.
S: Great. Is there anything else you would like to say about this picture?
U: No.S: Shall we move on to the next photo?
U: Yes, please.
(A new picture shows up)
S: Who is the one on the right?
U: That's John again.
(Move picture back to the previous one)
S: Is this the same John you mentioned in this picture?
U: Yes.
(Move picture back to the current one)
S: The one on the left of John is?
U: Can you read me some BBC news please?
(A newspaper picture shows up on screen)
S: What kind of news would you like to hear? Three channels: Sports, Politics and business are available.
U: Business please.
S: (Starts reading news)
U: Ok, stop please.
S: Would you like to hear more news of Business?
U: No, thanks.
(Back to the photos)
S: The person on the left of John is?
……..

Figure 2: Log of a User Session (S=System, U=User)

3. Natural Language Understanding

The Companions, Natural Language Understanding module uses a number of components to analyse the dialogue in order to eventually guide the Dialogue Manager in its response. The initial tasks of the Senior Companion is to identify relationships between people and locations in the dialogue.

If we consider the relationships task, this is achieved by first using GATE (Cunningham et al., 1996) to analyse the dialogue and then make use of an extended version of Annie (Bontcheva et al. 2002) (Gate's Information Extraction system) to extract the semantic information. Annie's extensions are basically gazetteers used to help the system identify relationships within the text. After tokenising, tagging and finding chunks in the text, Annie is used to extract the relationships and named entities. To identify and disambiguate the bindings between the relationships and the named entities, the syntactic structure of the text is analysed. The output of this process is a triple specifying that;

"person X is related to person Y because of the relation Z."

Each utterance is also split into sentences which are then analysed using a Dialogue Act Tagger (Hardy et al., 2004).

All the relationships used by the system can be found in the Companions Relationships Ontology. The ontology not only defines a hierarchy of relationships and their properties but it also specifies rules such as; "Brother is Male". These rules are then used to verify the information obtained from the IE system mentioned earlier. When the information is verified, the ontology is populated with this information and the inference engine is used to extrapolate further relationships. This allows us to deduce facts such as;

"if person X has a father Y and a Y has a brother Z, then we can infer that person X has an uncle Z while person Z has a nephew X."

When all the information in the utterance has been analysed, the resulting information is passed over to the Dialogue Manager.

The task of recognizing Locations follows a similar approach. First Named Entities are identified by Annie and information of the sort "person X travelled to location Y" is extracted. Then the Companions Locations Ontology is used to infer further information. This ontology contains information about Continents, Countries, Regions, etc. It even goes into the details of places of interest and things to do. The ontology has been populated with real information extracted from various online sources found all over the web. The information contained within this ontology will help the Dialogue Manager not only give a geographical position to places but also perform a number of generalisations or specialisations such as;

"if person X visited Venice, the system will know (from the ontology) about Gondola rides and can ask him if he took one. On the other hand, the system, knowing that Venice is in the Veneto region (from the ontology) can ask the person if he visited another place in the same region such as Verona."

Once again, when the information has been analysed, it is then passed over to the Dialogue Manager for further processing.

4. The Dialogue Manager

The Senior Companion Dialogue Manager is a multiagent system [Figure3] composed of two main agent types: behavior and control agents.
Behavior Agents embody a single conversational or operational task of the system, such as **"discover user name"**, **"chat_about_photo"**, **"talk about event"** and **"read news".** They are implemented as ATN's (Woods, W. 1970), as in the system designed for the COMIC project (Catizone, Setzer and Wilks, 2003). These ATNs are called Dialogue Action Forms (DAFs) and are used to exploit the IE information from the NLU module through the Indexing Terms component of the DM architecture (Figure 3).
Control agents are used to determine which behavior agent will run at each time step.
Each behavior is tagged at design time with a set of terms that characterize it. These terms can be restrictions on domain properties (such as time>18:00), keywords, parts-of-speech tags, named entities, or syntactic dependencies. Each behavior in the system has a unique key, formed by the set of its terms.
The dialogue manager uses the information about the system status and the input to make a set of indexing terms. It matches these indexing terms to the keys of each behavior and selects the behavior for execution that most closely matches the current indexing terms at each time step. A behavior, when selected for execution, keep the values of the indexing terms used for its selection – it is then called an instantiated behavior. These terms provide a partial context for the behavior.
Figure 3 shows the Dialogue Management System. Dashed arrows indicate the direction of broadcast messages. Full arrows indicate components that directly modify others. Full connections that are not arrows show just association.

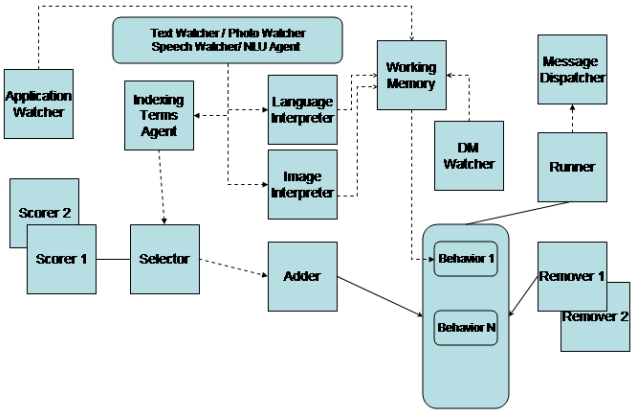

Figure 3: Agents of the Senior Companion Dialogue Manager

The *Adder* is the agent that takes care of the decision of what to do when a new behavior is selected for execution, as discussed in the previous paragraph. In this system it checks the stack for an interrupted behavior identical to the one to be stacked, and if it is the case, remove it from the stack and re-pushes it to the top. Otherwise it just pushes the new behavior. A behavior, when selected for execution, keep the values of the indexing terms used for its selection – it is then called an instantiated behavior. These terms provide a partial context for the behavior.
The *Remover* takes care of a problem that we have not discussed so far: how to perceive and decide when a conversational behavior is no longer relevant and what to do when it happens? Examples could be conversations that

were interrupted for so long that the user is no longer interested in them, or conversations that were subsumed by other conversations. The current system has two behavior *Remover* agents, one that removes behaviors that are inactive for a time longer than a constant and another that removes behaviors that get pushed down the stack beyond a certain depth.

The *Working Memory* (WM) is not as passive as the name might suggest. Besides keeping predicates and objects that correspond to the knowledge of interest to the behavior agents, is has three active roles: it tries to keep its knowledge consistent, actively forgets old information, and automatically infers new information whenever new predicates and objects are added to it. As the other agents in the system, it notifies subscribed agents of changes in its state. The *Adder* registers the instantiated behavior to listen to events in the *Working Memory*, so that the behavior always has the latest percepts during its execution.

The low level percepts coming from the *Text Watcher*, *Photo Watcher*, *Speech Watcher* and *NLU Agent* are caught by the *Indexing Terms Agent*, the *Language Interpreter* and the *Image Interpreter.*

The *Language Interpreter* adds predicates and objects to the *Working Memory*, based on what is already there, the system background knowledge and the outputs of the *NLU Agent*, *Speech Watcher* and *Text Watcher*. It is the element that will be able to do anaphora resolution.

The *Image Interpreter* uses information from the watchers to populate the working memory in the same way as the Language Interpreter. One example is the addition of predicates that describe the relative positions of the people described in the photos. It has background knowledge about pictures and spatial relations.

The *Application Watcher* is an application and system specific agent that creates objects and predicates representing aspects of the system that are used in behavior selection and execution. It is the agent that might populate the working memory with system time information, for example.

The *Indexing Terms* agent uses the information of the Working Memory, the NLU, and the watchers to create indexing terms for behavior selection. This is the component of the DM that exploits the results of the Information Extraction. So, for example, if we know that Zoe and Octavia are daughters of the user Roberta, then we also know that they (Z&O) are sisters and we use this information to discuss features of the sisters with the user such as age, favourite pastimes, etc..

A *Scorer* agent, under request of a *Selector*, produces a list of scores, each corresponding to a particular view of the indexing terms of the behaviors available for selection. We may have scorers that focus only on full matches, scorers that use term expansion to assign partial scores, scorers that just consider system properties, etc. The main motivation for this was to allow experimentation with different scoring policies, and to being able to treat each term type individually. The present system uses just full match scorers, one for system properties and one for keywords.

The *Selector* agent decides which behavior, if any, will be selected for addition whenever it receives new indexing terms. It calls the available scorers and uses a defined algorithm to combine them. Currently we select the highest scoring behavior considering the sum of all *Scorers*. In the future we will investigate the incorporation of default preferences and preferences based on the content of the stack.

The *Dialogue Manager Watcher* (DM Watcher) monitors the events inside the dialogue manager. It populates the Working Memory with predicates such as "NewUtteranceArrived(time)". The predicates and objects of the Dialogue Manager are used in operations of finer grained dialogue control and repair, usually carried out by specialized behaviors (an example would be "clarify last question").

A behavior in our system is ultimately implemented by an augmented finite-state machine(more specifically an augmented transition network (Woods, W., 1970)). Any action or check is performed by sending a message and receiving an acknowledgement (the FSM may ignore the acknowledgement, if it is not crucial)

The *Behavior Runner* (BR) is the agent that actually drives behavior execution, telling a stacked behavior when to be active and when to wait. It won't stop a behavior in the middle of a transition or action though, so the behavior always stops immediately after performing a transition or immediately before checking the transition conditions. The *Behavior Runner* is also the agent that removes behaviors that have finished their execution.

Finally, the *Message Dispatcher* is the agent that processes the messages from a behavior. It publishes the dialog system messages in a form amenable to the *Communication Agent* and *Application Domain Manager*.

5. Conclusion

We are at the start of the Companions project, but feel that our preliminary research indicates that the using IE tools and methods for improving Dialogue management is worth pursuing. We look forward to going further with this research and reporting our forthcoming results in the near future.

6. Acknowledgements

The authors' research was sponsored by the European Commission under EC grant IST-FP6-034434 (Companions).

7. References

Bontcheva et al., (2002). Bontcheva, K., Cunningham, H., Tablan, V., Maynard, D., and Saggion, H., Developing Reusable and Robust Language Processing Components for Information Systems using GATE. In 3rd International Workshop on Natural Language and

Information Systems (NLIS'2002), Aix-enProvence, France, 2002. IEEE Computer Society Press.

Catizone, R., Setzer, A., and Wilks, Y. (2003). "Multimodal Dialogue Management in the COMIC Project", Workshop on Dialogue Systems: interaction, adaptation and styles of management. (EACL)Budapest, Hungary.

Cunningham et al. (1997). Cunningham, H., Humphreys, K., Gaizauskas, R., Wilks, Y., GATE -- a TIPSTER based General Architecture for Text Engineering. In Proceedings of the TIPSTER Text Program (Phase III) 6 Month Workshop. DARPA, Morgan Kaufmann, California

Hardy et al. (2005). Hardy, H., A Biermann, R. Bryce Inouye, A. McKenzie, T. Strzalkowski, C. Ursu, N. Webb and M. Wu, *The AMITIES System: Data-Driven Techniques for Automated Dialogue*, in Speech Communication. Elsevier

William A. Woods (1970). Transition network grammars for natural language analysis. Commun ACM 13(10): 591-606

Young, S. (2006). Using POMDPs for Dialog Management, IEEE/ACL Workshop on Spoken Language Technology (SLT 2006), Aruba.