IRIS: a Chat-oriented Dialogue System based on the Vector Space Model

Rafael E. Banchs
Human Language Technology
Institute for Infocomm Research
Singapore 138632
`rembanchs@i2r.a-star.edu.sg`

Haizhou Li
Human Language Technology
Institute for Infocomm Research
Singapore 138632
`hli@i2r.a-star.edu.sg`

Abstract

This system demonstration paper presents IRIS (Informal Response Interactive System), a chat-oriented dialogue system based on the vector space model framework. The system belongs to the class of example-based dialogue systems and builds its chat capabilities on a dual search strategy over a large collection of dialogue samples. Additional strategies allowing for system adaptation and learning implemented over the same vector model space framework are also described and discussed.

1 Introduction

Dialogue systems have been gaining popularity recently as the demand for such kind of applications have increased in many different areas. Additionally, recent advances in other related language technologies such as speech recognition, discourse analysis and natural language understanding have made possible for dialogue systems to find practical applications that are commercially exploitable (Pieraccini *et al.*, 2009; Griol *et al.*, 2010).

From the application point of view, dialogue systems can be categorized into two major classes: task-oriented and chat-oriented. In the case of task-oriented dialogue systems, the main objective of such a system is to help the user to complete a task, which typically includes booking transportation or accommodation services, requesting specific information from a service facility, etc. (Busemann *et al.*, 1997; Seneff and Polifroni, 2000; Stallard, 2000). On the other hand, chat-oriented systems are not intended to help the user completing any specific task, but to provide a means for participating in a game, or just for chitchat or entertainment. Typical examples of chat-oriented dialogue systems are the so called chat bots (Weizenbaum, 1966; Ogura *et al.*, 2003, Wallis, 2010).

In this paper, we introduce IRIS (Informal Response Interactive System), a chat-oriented dialogue system that is based on the vector space model framework (Salton *et al.*, 1975; van Rijsbergen, 2005). From the operational point of view, IRIS belongs to the category of example-based dialogue systems (Murao *et al.*, 2003). Its dialogue strategy is supported by a large database of dialogues that is used to provide candidate responses to a given user input. The search for candidate responses is performed by computing the cosine similarity metric into the vector space model representation, in which each utterance in the dialogue database is represented by a vector.

Different from example-based question answering systems (Vicedo, 2002; Xue *et al.*, 2008), IRIS uses a dual search strategy. In addition to the current user input, which is compared with all existent utterances in the database, a vector representation of the current dialogue history is also compared with vector representations of full dialogues in the database. Such a dual search strategy allows for incorporating information about the dialogue context into the response selection process.

The rest of the paper is structured as follows. Section 2 presents the architecture of IRIS as well as provides a general description of the dataset that has been used for its implementation. Section 3 presents some illustrative examples of dialogues generated by IRIS, and Section 4 presents the main conclusions of this work.

2 The IRIS Implementation

In this section we first provide a detailed description of the IRIS architecture along with the most relevant issues behind its implementation. Then, we describe the specific dialogue dataset that supports the IRIS implementation.

2.1 Architecture

As already mentioned, IRIS architecture is heavily based on a vector space model framework, which includes a standard similarity search module from vector-based information retrieval systems (Salton and McGill, 1983). However, it also implements some additional modules that provide the system with capabilities for automatic chatting.

Figure 1 depicts a block diagram that illustrates the main modules in the IRIS architecture. As seen from the picture, the whole system comprises seven processing modules and three repositories.

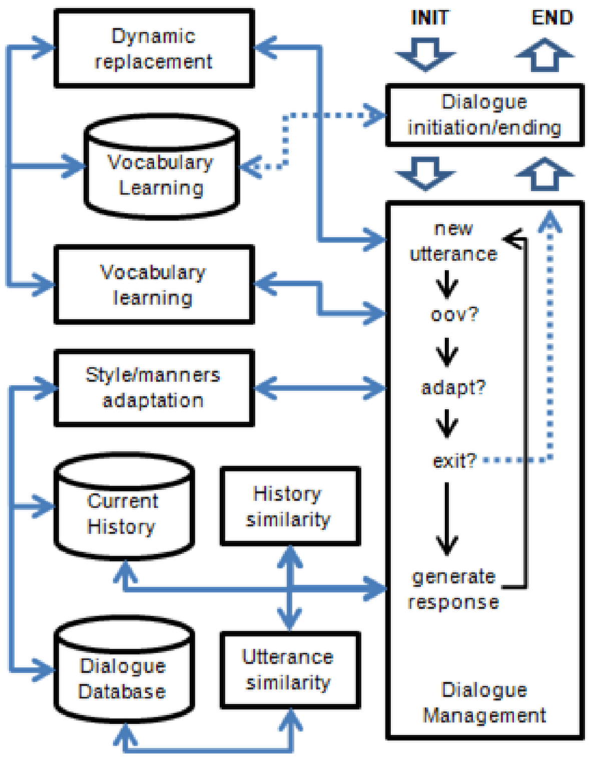

Figure 1: General block diagram for IRIS

The main operation of IRIS can be described as follows. When a new dialogue starts, the control of the dialogue is passed from the dialogue management module to the initiation/ending module. This module implements a two-state dialogue strategy which main objectives are: first, to greet the user and self-introduce IRIS and, second, to collect the name of the user. This module uses a basic parsing algorithm that is responsible for extracting the user's name from the provided input. The name is the first vocabulary term learned by IRIS, which is stored in the vocabulary learning repository.

Once the dialogue initiation has been concluded the dialogue management system gains back the control of the dialogue and initializes the current history vector. Two types of vector initializations are possible here. If the user is already know by IRIS, it will load the last stored dialogue history for that user; otherwise, IRIS will randomly select one dialogue history vector from the dialogue database. After this initialization, IRIS prompts the user for what he desires to do. From this moment, the example-based chat strategy starts.

For each new input from the user, the dialogue management module makes a series of actions that, after a decision process, can lead to different types of responses. In the first action, the dynamic replacement module searches for possible matches between the terms within the vocabulary learning repository and the input string. In a new dialogue, the only two terms know by IRIS are its own name and the user name. If any of this two terms are identified, they are automatically replaced by the placeholders <*self-name*> and <*other-name*>, respectively.

In the case of a mature dialogue, when there are more terms into the vocabulary learning repository, every term matched in the input is replaced by its corresponding definition stored in the vocabulary learning database.

Just after the dynamic replacement is conducted, tokenization and vectorization of the user input is carried out. During tokenization, an additional checking is conducted by the dialogue manager. It looks for any adaptation command that could be possibly inserted at the beginning of the user input. More details on adaptation commands will be given when describing the style/manner adaptation module. Immediately after tokenization, unknown vocabulary terms (OOVs) are identified. IRIS will consider as OOV any term that is not contained in either the dialogue or vocabulary learning databases. In case an OOV is identified, a set of heuristics (aiming at avoiding confusing misspellings with OOVs) are applied to decide whether IRIS should ask the user for the meaning of such a term.

If IRIS decides to ask for the meaning of the term, the control of the dialogue is passed to the vocabulary learning module which is responsible for collecting the meaning of the given term from the user or, alternatively, from an external source of information. Once the definition is collected and validated, it is stored along with the OOV term into the vocabulary learning repository. After completing a learning cycle, IRIS acknowledges the user about having "understood" the meaning of the term and control is passed back to the dialogue management module, which waits for a new user input.

If IRIS decides not to ask for the meaning of the OOV term, or if no OOV term has been identified, vectorization of the user input is completed by the vector similarity modules and similarity scores are computed for retrieving best matches from the dialogue database. Two different similarity scores are actually used by IRIS. The first score is applied at the utterance level. It computes the cosine similarities between the current user input vector and all single utterances stored in the database. This score is used for retrieving a large amount of candidate utterances from the dialogue database, generally between 50 and 100, depending on the absolute value of the associated scores.

The second score is computed over history vectors. The current dialogue history, which is available from the current history repository, includes all utterances interchanged by the current user and IRIS. In other to facilitate possible topic changes along the dialogue evolution, a damping or "forgetting" factor is used for giving more importance to the most recent utterances in the dialogue history. A single vector representation is then computed for the currently updated dialogue history after applying the damping factor. The cosine similarity between this vector and the vector representations for each full dialogue stored in the dialogue database are computed and used along with the utterance-level score for generating a final rank of candidate utterances. A log-linear combination scheme is used for combining the two scores. The dialogue management module randomly selects one of the top ranked utterances and prompts back to the user the corresponding reply (from the dialogue database) to the wining utterance.

Just immediately before prompting back the response to the user, the dynamic replacement module performs an inverse operation for replacing the two placeholders <*self-name*> and <*other-name*>, in case they occur in the response, by their actual values.

The final action taken by IRIS is related to the style/manner adaptation module. For this action to take place the user has to include one of three possible adaptations commands at the beginning of her/his new turn. The three adaptation commands recognized by IRIS are: ban (*), reinforce (+), and discourage (–). By using any of these three characters as the first character in the new turn, the user is requesting IRIS to modify the vector space representation of the previous selected response as follows:

- Ban (*): IRIS will mark its last response as a prohibited response and will not show such response ever again.
- Reinforce (+): IRIS will pull the vector space representation of its last selected utterance towards the vector space representation of the previous user turn, so that the probability of generating the same response given a similar user input will be increased.
- Discourage (–): IRIS will push the vector space representation of its last selected utterance apart from the vector space representation of the previous user turn, so that the probability of generating the same response given a similar user input will be decreased.

2.2 Dialogue Data Collection

For the current implementation of IRIS, a subset of the Movie-DiC dialogue data collection has been used (Banchs, 2012). Movie-DiC is a dialogue corpus that has been extracted from movie scripts which are freely available at The Internet Movie Script Data Collection (http://www.imsdb.com/). In this subsection, we present a brief description on the specific data subset used for the implementation of IRIS, as well as we briefly review the process followed for collecting the data and extracting the dialogues.

First of all, dialogues have to be identified and parsed from the collected html files. Three basic elements are extracted from the scripts: speakers, utterances and context. The speaker and utterance elements contain information about the characters who speak and what they said at each dialogue turn. On the other hand, context elements contain all the additional information (explanations and descriptions) appearing in the scripts.

The extracted dialogues are stored into a data structure such that the information about turn sequences within the dialogues and dialogue sequences within the scripts are preserved.

Some post-processing is also necessary to filter out and/or repair the most common parsing errors occurring during the dialogue extraction phase. Some of these errors include: bad script formatting, same-speaker turn continuations, explanatory notes inserted within the turns, misspelling of names in the speaker headers, changes in the encoding format, etc.

The final dialogue collection used in the IRIS implementation consists of dialogues from 153 movie scripts, mainly belonging to the comedy, action and family genres. Table 1 summarizes the main statistics of the resulting dataset.

Total number of movie scripts	153
Total number of dialogues	24,265
Total number of speaker turns	159,182
Average amount of dialogues per movie	158.59
Average amount of turns per dialogue	6.56
Total number of running words	1,670,879
Overall vocabulary size	34,903

Table 2: Main statistics of the movie dialogue dataset used in the IRIS implementation

For each turn in the dialogue collection, a vector space model representation was constructed. For this, the standard bag-of-words weighting scheme known as TF-IDF was used (Spark, 1972; Salton and Buckley, 1988).

Before performing the vectorization, word tokenization was conducted. In this step, all punctuation marks were removed, with the exception of the question "?" and exclamation "!" marks. Similarly, all other non-alphanumeric characters occurring in the utterances were removed as well. Also during the tokenization phase, all self-references to current speaker names in the utterances were replaced by the <*self-name*> placeholder, as well as all references to the names of other speakers participating in the same dialogue were replaced by the <*other-name*> place-holder.

Finally, a vector space model representation was also computed for each full dialogue in the collection. For this bag-of-words model at the dialogue level, both utterance and context information were taken into account. Again, the TF-IDF weighting scheme was used.

3 Some Dialogue Examples

In this section we show some real examples of interactions between IRIS and human users. First, we present some interesting examples of good performance, as well as illustrate some of the learning capabilities of IRIS. Then, we present some of the common failures which identify specific points of attention for further improvements.

3.1 Good Performance Examples

Our first example illustrates the beginning of a typical chat session between IRIS and a new user. This example is depicted in Table 2.

#	Spkr.	Turn
1	IRIS	Hi, I'm IRIS… What's your name?
2	USER	Hi Iris, I am Peter
3	IRIS	Nice name Peter... What do you want to talk about?
4	USER	Sports?
5	IRIS	How come in all that time I was trying to get you a date, you never asked me out?
6	USER	ok, do you want to go for dinner tonight?
7	IRIS	Yes.
8	USER	What kind of food do you like?
9	IRIS	Anything but guts and mushrooms.
10	USER	Ok pick you up at seven!
11	IRIS	Ok. Pick me up at seven.

Table 2: Beginning of a chat session between IRIS and a new user

For the dialogue depicted in Table 2, turn numbers 1, 2 and 3 are processed by the dialogue intiation/ending module. The example-based dialogue management strategy starts from turn 4 onwards. Notice that as far as this is a new user, not previous dialogue history exists, so in this case a random history vector has been selected and instead of focusing in the *sports* topic suggested by the user, IRIS "takes the initiative" of asking for a date.

In our second example, which is presented in Table 3, we illustrate the beginning of a typical chat session between IRIS and a returning user. For this particular user, her last interaction with IRIS was about sports.

Similar to the previous example, turn 1 is processed by the dialogue intiation/ending module and the example-based dialogue management strategy starts from turn 2 onwards. In this particular case, IRIS is much more centered on the *sports* topic as this context information has been already provided by the stored dialogue history of the previous chat session with this particular user.

#	Spkr.	Turn
1	IRIS	Hey, Helen... welcome back! How are you doing?
2	USER	fine and you?
3	IRIS	Is there more?
4	USER	Yes, I finally went to that tennis match I told you the last time.
5	IRIS	The match!
6	USER	Yes, have you seen some sports recently?
7	IRIS	I hate sports.

Table 3: Beginning of a chat session between IRIS and a returning user

In our third example, which is presented in Table 4, we illustrate the learning of a new vocabulary word by IRIS. In this example, when the unknown term *paella* is detected, the control of the dialogue is passed to the vocabulary learning module, which takes care of turns 7, 8, 9 and 10.

#	Spkr.	Turn
1	USER	do you cook anything Iris?
2	IRIS	Eggs.
3	USER	ha ha ha!
4	IRIS	Ha-ha-ha! Ha-ha-ha!
5	USER	Can you cook anything else?
6	IRIS	Do you cook?
7	USER	Paella
8	IRIS	What the hell is that????
9	USER	It is a Spanish food. Yellow rice with some seafood on it.
10	IRIS	Ok, I see... thanks!
11	USER	So, do you want some paella today?
12	IRIS	Well, it beats working in a seafood restaurant...

Table 4: Chat segment in which IRIS learns the new vocabulary term *paella*

Notice that when the user asks IRIS about having some *paella* today, IRIS is already able to associate it with *seafood* as it was stated in the user's provided definition. The process actually occurs as follows: after tokenization, but before vectorization of the user input in turn 11, the dynamic replacement module substitutes the term *paella* by its definition, which has been previously stored in the vocabulary learning repository when turn 9 was processed. The actual user input that is finally vectorized in turn 11 is the following one: *so do you want some it is a spanish food yellow rice with some seafood on it today ?*, which is the utterance used by IRIS to retrieve and select the response it provides in turn 12.

3.2 Common Failure Examples

In this subsection we focus our attention in the most common failures exhibited by IRIS. Some of these failures put in evidence specific points of attention that should be taken into account for further improvements of the system.

Our first example illustrates the problem of IRIS lack of consistency in issues for which consistent answers are required. Two specific chat segments in which IRIS provides inconsistent responses are presented in Table 5.

#	Spkr.	Turn
1	USER	how old are you Iris?
2	IRIS	I'm thirteen, how old are you?
3	USER	I am 40
4	IRIS	Only I did better... 40 crates a day.
5	USER	how old are you Iris?
6	IRIS	Ten.
#	**Spkr.**	**Turn**
1	USER	Yes, have you seen some sports recently?
2	IRIS	I hate sports.
3	USER	Really, what about soccer?
4	IRIS	I'd sooner watch soccer.
5	USER	what do you prefer soccer or football?
6	IRIS	Football is my life.

Table 5: IRIS incapability to being consistent

The first example presented in Table 5 constitutes a serious consistency problem. In this case IRIS has reported two different ages in the same chat session. The second case, although not so serious as the previous one, also constitutes a consistency failure. In this case IRIS states *Football is my life* just two turns after saying *I hate sports*.

Our second example, which is presented in Table 6, illustrates a problem derived from the noise that is still present in the dataset.

#	Spkr.	Turn
1	IRIS	You watching the ballgame?
2	USER	No
3	IRIS	Bianca stares at him for a moment.

Table 6: Example of noise in the dialogue dataset

In the particular example illustrated in Table 6, as seen from turn 3, a context element has been mistakenly stored in the data collection as an utterance during the dataset preparation phase. Several problems similar to this one, which are related to noise in the database, have been detected already. To tackle this problem we need to refine the parsing and post-processing algorithms used during the dialogue dataset construction phase.

4 Conclusions and Future Work

In this paper, we have presented IRIS (Informal Response Interactive System), a chat-oriented dialogue system that is based on the vector space model framework. The system belongs to the class of example-based dialogue systems and builds its chat capabilities on a dual search strategy over a large collection of movie dialogues.

Additional strategies allowing for system adaptation and learning have been also implemented over the same vector space model framework. More specifically, IRIS is capable of learning new vocabulary terms and semantically relating them to previous knowledge, as well as adapting its dialogue decisions to some stated user preferences.

We have also described the main characteristics of the architecture of IRIS and the most important functions performed by each of its constituent modules. Finally, we have provided some examples of good chat performance and some examples of the common failures exhibited by IRIS.

As future work, we intend to improve IRIS performance by addressing some of the already identified common failures. Similarly, we intend to augment IRIS chatting capabilities by extending the size of the current dialogue database and integrating a strategy for group chatting.

Acknowledgments

The authors would like to thank the Institute for Infocomm Research for its support and permission to publish this work.

References

Banchs R E (2012) Movie-DiC: a movie dialogue corpus for research and development. In Proc. of the 50th Annual Meeting of the ACL.

Busemann S, Declerck T, Diagne A, Dini L, Klein J, Schmeier S (1997) Natural language dialogue service for appointment scheduling agents. In Proc. of the 5th Conference on Applied NLP, pp 25-32.

Griol D, Callejas Z, Lopez-Cozar R (2010) Statistical dialog management methodologies for real applications. In Proc. of SIGDIAL'10, pp 269-272.

Murao H, Kawaguchi N, Matsubara S, Yamaguchi Y, Inagaki Y (2003) Example-based spoken dialogue system using WOZ system log. In Proc. of the 4th SIGDIAL, pp 140-148.

Ogura K, Masuda T, Ishizaki M (2003) Building a new Internet chat system for sharing timing information. In Proc. of the 4th SIGDIAL, pp 97-104.

Pieraccini R, Suendermann D, Dayanidhi K, Liscombe J (2009) Are we there yet? Research in commercial spoken dialog systems. In Proc. of TSD'09, pp 3-13.

Salton G, Wong A, Yang C (1975) A vector space model for automatic indexing. Communications of the ACM 18(11):613-620.

Salton G, McGill M (1983) Introduction to modern information retrieval. McGraw-Hill.

Salton G, Buckley C (1988) Term-weighting approaches in automatic text retrieval. Information Processing & Management 24(5):513-523

Seneff S, Polifroni J (2000) Dialogue management in the Mercury flight reservation system. In Proc. of the ANLP-NAACL 2000 Workshop on Conversational Systems, pp 11-16.

Spark K (1972) A statistical interpretation of term specificity and its application in retrieval. Journal of Documentation 28(1):11-21

Stallard D (2000) Talk'n'travel: a conversational system for air travel planning. In Proc. of the 6th Conference on Applied NLP, pp 68-75.

van Rijsbergen C (2005) A probabilistic logic for information retrieval. In Advances in Information Retrieval, Lecture Notes in Computer Science 3408:1-6.

Vicedo J (2002) SEMQA: A semantic model applied to question answering systems. PhD Thesis, University of Alicante.

Wallis P (2010) A robot in the kitchen. In Proceedings of the ACL 2010 Workshop on Companionable Dialogue Systems, pp 25-30.

Weizenbaum J (1966) ELIZA – A computer program for the study of natural language communication between man and machine. Communications of the ACM 9(1):36-45.

Xue X, Jeon J, Croft W (2008) Retrieval models for question and answer archives. In Proc. of the 31st Annual International ACM SIGIR Conference on R&D in Information Retrieval, pp 475-482.