Adaptive Referring Expression Generation in Spoken Dialogue Systems: Evaluation with Real Users

Srinivasan Janarthanam
School of Informatics
University of Edinburgh
s.janarthanam@ed.ac.uk

Oliver Lemon
Interaction Lab
Mathematics and Computer Science
Heriot-Watt University
o.lemon@hw.ac.uk

Abstract

We present new results from a real-user evaluation of a data-driven approach to learning user-adaptive referring expression generation (REG) policies for spoken dialogue systems. Referring expressions can be difficult to understand in technical domains where users may not know the technical 'jargon' names of the domain entities. In such cases, dialogue systems must be able to model the user's (lexical) domain knowledge and use appropriate referring expressions. We present a reinforcement learning (RL) framework in which the system learns REG policies which can adapt to unknown users online. For real users of such a system, we show that in comparison to an adaptive hand-coded baseline policy, the learned policy performs significantly better, with a 20.8% average increase in adaptation accuracy, 12.6% decrease in time taken, and a 15.1% increase in task completion rate. The learned policy also has a significantly better subjective rating from users. This is because the learned policies adapt online to changing evidence about the user's domain expertise. We also discuss the issue of evaluation in simulation versus evaluation with real users.

1 Introduction

We present new results from an evaluation with real users, for a reinforcement learning (Sutton and Barto, 1998) framework to learn user-adaptive referring expression generation policies from data-driven user simulations. Such a policy allows the system to choose appropriate expressions to refer to domain entities in a dialogue setting. For instance, in a technical support conversation, the system could choose to use more technical terms with an expert user, or to use more descriptive and general expressions with novice users, and a mix of the two with intermediate users of various sorts (see examples in Table 1).

Jargon: Please plug one end of the `broadband cable` into the `broadband filter`.
Descriptive: Please plug one end of the `thin white cable with grey ends` into the `small white box`.

Table 1: Referring expression examples for 2 entities (from the corpus)

In natural human-human conversations, dialogue partners learn about each other and adapt their language to suit their domain expertise (Issacs and Clark, 1987). This kind of adaptation is called `Alignment through Audience Design` (Clark and Murphy, 1982; Bell, 1984). We assume that users are mostly unknown to the system and therefore that a spoken dialogue system (SDS) must be capable of observing the user's dialogue behaviour, modelling his/her domain knowledge, and adapting accordingly, just like human interlocutors. Therefore unlike systems that use static user models, our system has to dynamically model the user's domain knowledge in order to adapt during the conversation.

We present a corpus-driven framework for learning a user-adaptive REG policy from a small corpus of non-adaptive human-machine interaction. We show that the learned policy performs better than a simple hand-coded adaptive policy in terms of accuracy of adaptation, dialogue time and task completion rate when evaluated with real users in a wizarded study.

In section 2, we present some of the related work. Section 3 and section 4 describe the dialogue system framework and the user simulation model. In section 5, we present the training and in section 6, we present the evaluation for different REG policies with real users.

2 Related work

Rule-based and supervised learning approaches have been proposed to learn and adapt during conversations dynamically. Such systems learn from a user at the start and later adapt to the domain knowledge of the user. However, they either require expensive expert knowledge resources to hand-code the inference rules (Cawsey, 1993) or a large corpus of expert-layperson interaction from which adaptive strategies can be learned and modelled, using methods such as Bayesian networks (Akiba and Tanaka, 1994). In contrast, we present an approach that learns in the absence of these expensive resources. It is also not clear how supervised approaches choose between when to seek more information and when to adapt. In this study, we show that using reinforcement learning this decision is learned automatically.

Reinforcement Learning (RL) has been successfully used for learning dialogue management policies since (Levin et al., 1997). The learned policies allow the dialogue manager to optimally choose appropriate dialogue acts such as instructions, confirmation requests, and so on, under uncertain noise or other environment conditions. There have been recent efforts to learn information presentation and recommendation strategies using reinforcement learning (Hernandez et al., 2003; Rieser and Lemon, 2009; Rieser and Lemon, 2010), and joint optimisation of Dialogue Management and NLG using hierarchical RL has been proposed by (Lemon, 2010). In addition, we present a framework to learn to choose appropriate referring expressions based on a user's domain knowledge. Following a proof-of-concept study using a hand-coded rule-based user simulation (Janarthanam and Lemon, 2009c), we previously showed that adaptive REG policies can be learned using an RL framework with data-driven user simulations and that such policies perform better than simple hand-coded policies (Janarthanam and Lemon, 2010).

3 The Dialogue System

In this section, we describe the different modules of the dialogue system. The interaction between the different modules is shown in figure 1 (in learning mode). The dialogue system presents the user with instructions to setup a broadband connection at home. In the Wizard of Oz setup, the system and the user interact using speech. However, in our machine learning setup, they interact at the abstract level of dialogue actions and referring expressions. Our objective is to learn to choose the appropriate referring expressions to refer to the domain entities in the instructions.

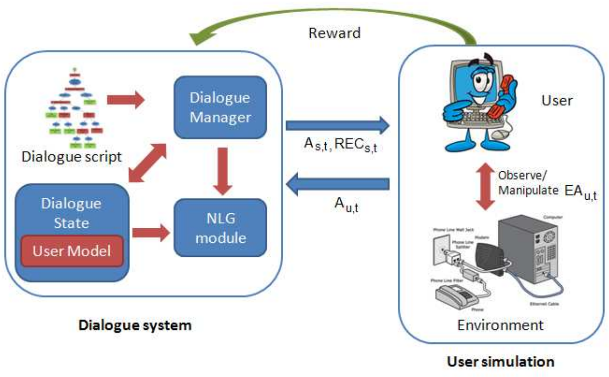

Figure 1: System User Interaction (learning)

3.1 Dialogue Manager

The dialogue manager identifies the next dialogue act ($A_{s,t}$ where t denotes turn, s denotes system) to give to the user based on the dialogue management policy π_{dm}. The dialogue management is coded in the form of a finite state machine. In this dialogue task, the system provides instructions to either observe or manipulate the environment. When users ask for clarifications on referring expressions, the system clarifies ($provide\_clar$) by giving information to enable the user to associate the expression with the intended referent. When the user responds in any other way, the instruction is simply repeated. The dialogue manager is also responsible for updating and managing the system state $S_{s,t}$ (see section 3.2). The system interacts with the user by passing both the system action $A_{s,t}$ and the referring expressions $REC_{s,t}$ (see section 3.3).

3.2 The dialogue state

The dialogue state $S_{s,t}$ is a set of variables that represent the current state of the conversation. In our study, in addition to maintaining an overall dialogue state, the system maintains a user model $UM_{s,t}$ which records the initial domain knowledge of the user. It is a dynamic model that starts with a state where the system does not have any knowledge about the user. Since the model is updated according to the user's behaviour, it may be inaccurate if the user's behaviour is itself uncertain. Hence, the user model used in this system is not always an accurate model of the user's knowledge and reflects a level of uncertainty about the user.

Each jargon referring expression `x` is represented by a three-valued variable in the dialogue state: `user_knows_x`. The three values that each variable takes are `yes`, `no`, `not_sure`. The variables are updated using a simple user model update algorithm after the user's response each turn. Initially each variable is set to `not_sure`. If the user responds to an instruction containing the referring expression `x` with a clarification request, then `user_knows_x` is set to `no`. Similarly, if the user responds with appropriate information to the system's instruction, the dialogue manager sets `user_knows_x` is set to `yes`. Only the user's initial knowledge is recorded. This is based on the assumption that an estimate of the user's initial knowledge helps to predict the user's knowledge of the rest of the referring expressions.

3.3 REG module

The REG module is a part of the NLG module whose task is to identify the list of domain entities to be referred to and to choose the appropriate referring expression for each of the domain entities for each given dialogue act. In this study, we focus only on the production of appropriate referring expressions to refer to domain entities mentioned in the dialogue act. It chooses between the two types of referring expressions - jargon and descriptive. For example, the domain entity *broadband_filter* can be referred to using the jargon expression "broadband filter" or using the descriptive expression "small white box"[1]. Although adaptation is the primary goal, it should be noted that in order to get an idea of the user the system is dealing with, it needs to seek information using jargon expressions.

The REG module operates in two modes - learning and evaluation. In the learning mode, the REG module is the learning agent. The REG module learns to associate dialogue states with optimal referring expressions. This is represented by a REG policy $\pi_{reg} : UM_{s,t} \rightarrow REC_{s,t}$, which maps the states of the dialogue (user model) to optimal referring expressions. The referring expression choices $REC_{s,t}$ is a set of pairs identifying the referent R and the type of expression T used in the current system utterance. For instance, the pair $(\textit{broadband\_filter}, desc)$ represents the descriptive expression "small white box".

$$REC_{s,t} = \{(R_1, T_1), ..., (R_n, T_n)\}$$

In the evaluation mode, a trained REG policy interacts with unknown users. It consults the learned policy π_{reg} to choose the referring expressions based on the current user model.

4 User Simulations

In this section, we present user simulation models that simulate the dialogue behaviour of a real human user. Several user simulation models have been proposed for use in reinforcement learning of dialogue policies (Georgila et al., 2005; Schatzmann et al., 2006; Schatzmann et al., 2007; Ai and Litman, 2007). However, they are suited only for learning dialogue management policies, and not natural language generation policies. In particular, our model is the first to be sensitive to a system's choices of referring expressions. Earlier, we presented a two-tier simulation trained on data precisely for REG policy learning (Janarthanam and Lemon, 2009a). However, it is not suited for training on small corpus like the one we have at our disposal. In contrast to the earlier model, we now condition the clarification requests on the referent class rather than the referent itself to handle the data sparsity problem.

4.1 Corpus-driven action selection model

The user simulation (US) receives the system action $A_{s,t}$ and its referring expression choices $REC_{s,t}$ at each turn. The US responds with a user action $A_{u,t}$ (u denoting user). This can either be a clarification request (cr) or an instruction response (ir). The US produces a clarification request cr based on the class of the referent $C(R_i)$, type of the referring expression T_i, and the current domain knowledge of the user for the referring expression $DK_{u,t}(R_i, T_i)$. Domain entities whose jargon expressions raised clarification requests in the corpus were listed and those that had more than the mean number of clarification requests were classified as `difficult` and others as `easy` entities (for example, *power_adaptor* is `easy` - all

[1]We will use italicised forms to represent the domain entities (e.g. *broadband_filter*) and double quotes to represent the referring expressions (e.g. "broadband filter").

users understood this expression, *broadband_filter* is `difficult`). Clarification requests are produced using the following model.

$$P(A_{u,t} = cr(R_i, T_i)|C(R_i), T_i, DK_{u,t}(R_i, T_i)) \\ \text{where } (R_i, T_i) \in REC_{s,t}$$

One should note that the actual literal expression is not used in the transaction. Only the entity that it is referring to (R_i) and its type (T_i) are used. However, the above model simulates the process of interpreting and resolving the expression and identifying the domain entity of interest in the instruction. The user identification of the entity is signified when there is no clarification request produced (i.e. $A_{u,t} = none$). When no clarification request is produced, the environment action $EA_{u,t}$ is generated using the following model.

$$P(EA_{u,t}|A_{s,t}) \text{ if } A_{u,t}! = cr(R_i, T_i)$$

Finally, the user action is an instruction response which is determined by the system action $A_{s,t}$. Instruction responses can be either $provide\_info$, $acknowledgement$ or $other$ based on the system's instruction.

$$P(A_{u,t} = ir|EA_{u,t}, A_{s,t})$$

All the above models were trained on our corpus data using *maximum likelihood estimation* and smoothed using a variant of *Witten-Bell discounting*. The corpus contained dialogues between a non-adaptive dialogue system and real users. According to the data, clarification requests are much more likely when jargon expressions are used to refer to the referents that belong to the `difficult` class and which the user doesn't know about. When the system uses expressions that the user knows, the user generally responds to the instruction given by the system.

4.2 User Domain knowledge

The user domain knowledge is initially set to one of several models at the start of every conversation. The models range from novices to experts which were identified from the corpus using `k-means` clustering. A novice user knows only "power adaptor", an expert knows all the jargon expressions and intermediate users know some. We assume that users can interpret the descriptive expressions and resolve their references. Therefore, they are not explicitly represented. We only code the user's knowledge of jargon expressions using boolean variables representing whether the user knows the expression or not.

4.3 Corpus

We trained the action selection model on a small corpus of 12 non-adaptive dialogues between real users and a dialogue system. There were six dialogues in which users interacted with a system using just jargon expressions and six with a system using descriptive expressions. For more discussions on our user simulation models and the corpus, please refer to (Janarthanam and Lemon, 2009b; Janarthanam and Lemon, 2009a; Janarthanam and Lemon, 2010).

5 Training

The REG module was trained (operated in learning mode) using the above simulations to learn REG policies that select referring expressions based on the user expertise in the domain. In this section, we discuss how to code the learning agent's goals as reward. We then discuss how the reward function is used to train the learning agent.

5.1 Reward function

We designed a reward function for the goal of adapting to each user's domain knowledge. We present the Adaptation Accuracy score (AA) that calculates how accurately the agent chose the appropriate expressions for each referent r, with respect to the user's knowledge. So, when the user knows the jargon expression for r, the appropriate expression to use is jargon, and if s/he doesn't know the jargon, a descriptive expression is appropriate. Although the user's domain knowledge is dynamically changing due to learning, we base appropriateness on the initial state, because our objective is to adapt to the initial state of the user $DK_{u,initial}$. However, in reality, designers might want their system to account for user's changing knowledge as well. We calculate accuracy per referent RA_r and then calculate the overall mean adaptation accuracy (AA) over all referents as shown below.

$$RA_r = \frac{\#(appropriate\_expressions(r))}{\#(instances(r))}$$

$$AdaptationAccuracyAA = \frac{1}{\#(r)}\Sigma_r RA_r$$

5.2 Learning

The REG module was trained in learning mode using the above reward function using the SHARSHA reinforcement learning algorithm (with linear function approximation) (Shapiro and Langley, 2002). This is a hierarchical variant of SARSA, which is an on-policy learning algorithm that updates the current behaviour policy (see (Sutton and Barto, 1998)). The training produced approx. 5000 dialogues. The user simulation was calibrated to produce three types of users: Novice, Intermediate and Expert, randomly but with equal probability.

Initially, the REG policy chooses randomly between the referring expression types for each domain entity in the system utterance, irrespective of the user model state. Once the referring expressions are chosen, the system presents the user simulation with both the dialogue act and referring expression choices. The choice of referring expression affects the user's dialogue behaviour. For instance, choosing a jargon expression could evoke a clarification request from the user, based on which, the dialogue manager updates the internal user model ($UM_{s,t}$) with the new information that the user is ignorant of the particular expression. It should be noted that using a jargon expression is an *information seeking* move which enables the REG module to estimate the user's knowledge level. The same process is repeated for every dialogue instruction. At the end of the dialogue, the system is rewarded based on its choices of referring expressions. If the system chooses jargon expressions for novice users or descriptive expressions for expert users, penalties are incurred and if the system chooses REs appropriately, the reward is high. On the one hand, those actions that fetch more reward are reinforced, and on the other hand, the agent tries out new state-action combinations to explore the possibility of greater rewards. Over time, it stops exploring new state-action combinations and exploits those actions that contribute to higher reward. The REG module learns to choose the appropriate referring expressions based on the user model in order to maximize the overall adaptation accuracy. Figure 2 shows how the agent learns using the data-driven (**Learned DS**) during training. It can be seen in the figure 2 that towards the end the curve plateaus, signifying that learning has converged.

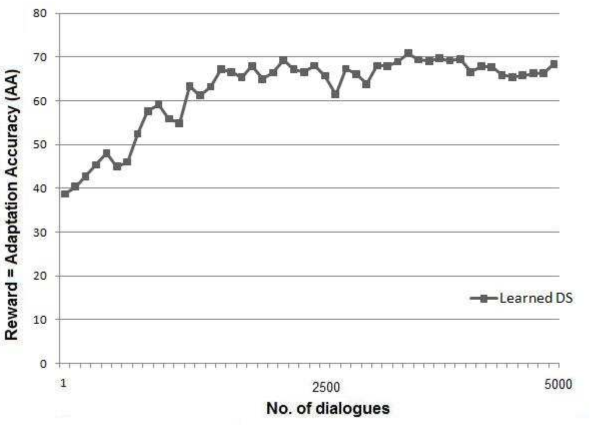

Figure 2: Learning curve - Training

6 Evaluation

In this section, we present the details of the evaluation process, the baseline policy, the metrics used, and the results. In a recent study, we evaluated the learned policy and several hand-coded baselines with simulated users and found that the Learned-DS policy produced higher adaptation accuracy than other policies (Janarthanam and Lemon, 2010). An interesting issue for research in this area is to what extent evaluation results obtained in simulated environments transfer to evaluations with real users (Lemon et al., 2006).

6.1 Baseline system

In order to compare the performance of the learned policy with a baseline, a simple rule-based policy was built. This baseline was chosen because it performed better in simulation, compared to a variety of other baselines (Janarthanam and Lemon, 2010). It uses jargon for all referents by default and provides clarifications when requested. It exploits the user model in subsequent references after the user's knowledge of the expression has been set to either `yes` or `no`. Therefore, although it is a simple policy, it adapts to a certain extent ('locally'). We refer to this policy as the 'Jargon-adapt' policy. It should be noted that this policy was built in the absence of expert domain knowledge and/or an expert-layperson corpus.

6.2 Process

We evaluated the two policies with real users. 36 university students from different backgrounds (e.g. Arts, Humanities, Medicine and Engineering) participated in the evaluation. 17 users were given a system with Jargon-adapt policy and 19 users interacted with a system with Learned-DS policy. Each user was given a pre-task recognition test to record his/her initial domain knowledge. The experimenter read out a list of technical terms and the user was asked to point out to the domain entities laid out in front of them. They were then given one of the two systems - learned or baseline, to interact with. Following the system instructions, they then attempted to set up the broadband connection. When the dialogue had ended, the user was given a post-task test where the recognition test was repeated and their responses were recorded. The user's broadband connection setup was manually examined for task completion (i.e. the percentage of correct connections that they had made in their final set-up). The user was given the task completion results and was then given a user satisfaction questionnaire to evaluate the features of the system based on the conversation.

All users interacted with a wizarded system employing one of the two REG policies (see figure 3). The user's responses were intercepted by a human interpreter (or "wizard") and were annotated as dialogue acts, to which the automated dialogue manager responded with a system dialogue action (the dialogue policy was fixed). The wizards were not aware of the policy used by the system. The respective policies chose only the referring expressions to generate the system utterance for the given dialogue action. The system utterances were converted to speech by a speech synthesizer (Cereproc) and were played to the user.

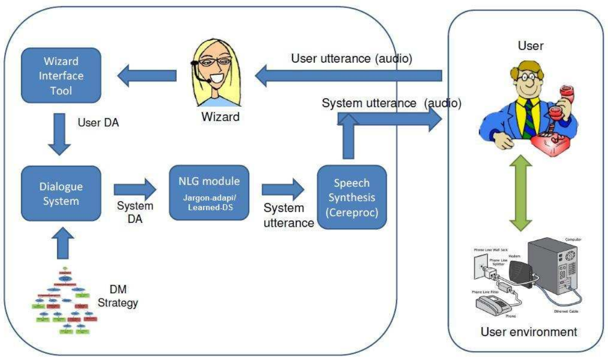

Figure 3: Wizarded Dialogue System

6.3 Metrics

In addition to the adaptation accuracy mentioned in section 5.1, we also measure other parameters from the conversation in order to show how learned adaptive policies compare with other policies on other dimensions. We also measure the learning effect on the users as (normalised) learning gain (LG) produced by using unknown jargon expressions. This is calculated using the pre- and post-test scores for the user domain knowledge (DK_u) as follows.

Metrics	Jargon-adapt	Learned-DS
AA	63.91	84.72 [**]
LG	0.59	0.61
DT	7.86	6.98 [*]
TC	84.7	99.8 [**]

[*] Statistical significance ($p < 0.05$).
[**] Statistical significance ($p < 0.001$).

Table 2: Evaluation with real users

$$Learning\ Gain\ LG = \frac{Post-Pre}{1-Pre}$$

Dialogue time (DT) is the actual time taken for the user to complete the task. We measured task completion (TC) by examining the user's broadband setup after the task was completed (i.e. the percentage of correct connections that they had made in their final set-up).

6.4 Results

We compare the performance of the two strategies on real users using objective parameters and subjective feedback scores. Tests for statistical significance were done using Mann-Whitney test for 2 independent samples (due to non-parametric nature of the data).

Table 2 presents the mean accuracy of adaptation (AA), learning gain (LG), dialogue time (DT), and task completion (TC), produced by the two strategies. The Learned-DS strategy produced more accurate adaptation than the Jargon-adapt strategy (p<0.001, U=9.0, r=-0.81). Higher accuracy of adaptation (AA) of the Learned-DS strategy translates to less dialogue time (U=73.0, p<0.05, r=-0.46) and higher task completion (U=47.5, p<0.001, r=-0.72) than the Jargon-adapt policy. However, there was no significant difference in learning gain (LG).

Table 3 presents how the users subjectively scored on a agreement scale of 1 to 4 (with 1 meaning 'strongly disagree'), different features of the system based on their conversations with the two different strategies. Users' feedback on different features of the systems were not very different from each other. However, users did feel that it was easier to identify domain objects with the Learned-DS strategy than the Jargon-adapt strategy (U=104.0, p<0.05, r=-0.34). To our knowledge, this is the first study to show a significant improvement in real user ratings for a learned policy in spoken dialogue systems (normally, objective metrics show an improvement, but not subjec-

Feedback questions	Jargon-adapt	Learned-DS
Q1. Quality of voice	3.11	3.36
Q2. Had to ask too many questions	2.23	1.89
Q3. System adapted very well	3.41	3.58
Q4. Easy to identify objects	2.94	3.37 *
Q5. Right amount of dialogue time	3.23	3.26
Q6. Learned useful terms	2.94	3.05
Q7. Conversation was easy	3.17	3.42
Q8. Future use	3.22	3.47

* Statistical significance ($p < 0.05$).

Table 3: Real user feedback

tive scores (Lemon et al., 2006)).

6.5 Analysis

The results show that the Learned-DS strategy is significantly better than the hand-coded Jargon-Adapt policy in terms of adaptation accuracy, dialogue time, and task completion rate. The initial knowledge of the users (mean pre-task recognition score) of the two groups were not significantly different from each other (Jargon-adapt = 7.33, Learned-DS = 7.45). Hence there is no bias on the user's pre-task score towards any strategy. While the Learned-DS system adapts well to its users globally, the Jargon-adapt system adapted only locally. This led to higher task completion rate and lower dialogue time.

The Learned-DS strategy enabled the system to adapt using the dependencies that it learned during the training phase. For instance, when the user asked for clarification on some referring expressions (e.g. "ethernet cable"), it used descriptive expressions for domain objects like ethernet light and ethernet socket. Such adaptation across referents enabled the Learned-DS strategy to score better than the Jargon-adapt strategy. Since the agent starts the conversation with no knowledge about the user, it learned to use information seeking moves (use jargon) at appropriate moments, although they may be inappropriate. But since it was trained to maximize the adaptation accuracy, the agent also learned to restrict such moves and start predicting the user's domain knowledge as soon as possible. By learning to trade-off between information-seeking and adaptation, the Learned-DS policy produced a higher adaptation with real users with different domain knowledge levels.

The users however did not generally rate the two policies differently. However, they did rate it (significantly) easier to identify objects when using the learned policy. For the other ratings, users seemed to be not able to recognize the nuances in the way the system adapted to them. They could have been satisfied with the fact that the system adapted better (Q3). This adaptation and the fact that the system offered help when the users were confused in interpreting the technical terms, could have led the users to score the system well in terms of future use (Q8), dialogue time (Q5), and ease of conversation (Q7), but in common with experiments in dialogue management (Lemon et al., 2006) it seems that users find it difficult to evaluate these improvements subjectively. The users were given only one of the two strategies and therefore were not in a position to compare the two strategies and judge which one is better. Results in table 3 lead us to conclude that perhaps users need to compare two or more strategies in order to judge the strategies better.

7 Conclusion

We presented new results from an evaluation with real users. In this study, we have shown that user-adaptive REG policies can be learned using an RL framework and data-driven user simulations. It learned to trade off between adaptive moves and information seeking moves automatically to maximize the overall adaptation accuracy. The learned policy started the conversation with information seeking moves, learned a little about the user, and started adapting dynamically as the conversation progressed. We also showed that the learned policy performs better than a reasonable hand-coded policy with real users in terms of accuracy of adaptation, dialogue time, task completion, and a subjective evaluation. Finally, this paper provides further evidence that evaluation results obtained in simulated environments can transfer reliably to evaluations with real users (Lemon et al., 2006).

Whether the learned policy would perform better than a hand-coded policy which was painstakingly crafted by a domain expert (or learned using supervised methods from an expert-layperson corpus) is an interesting question that needs further exploration. Also, it would also be interesting to make the learned policy account for the user's learning behaviour and adapt accordingly. We also believe that this framework can be extended to include other decisions in NLG besides REG (Dethlefs and Cuayahuitl, 2010).

Acknowledgements

The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement no. 216594 (CLASSiC project `www.classic-project.org`) and from the EPSRC, project no. EP/G069840/1.

References

H. Ai and D. Litman. 2007. Knowledge consistent user simulations for dialog systems. In *Proceedings of Interspeech 2007, Antwerp, Belgium*.

T. Akiba and H. Tanaka. 1994. A Bayesian approach for User Modelling in Dialogue Systems. In *Proceedings of the 15th conference on Computational Linguistics - Volume 2, Kyoto*.

A. Bell. 1984. Language style as audience design. *Language in Society*, 13(2):145–204.

A. Cawsey. 1993. User Modelling in Interactive Explanations. *User Modeling and User-Adapted Interaction*, 3(3):221–247.

H. H. Clark and G. L. Murphy. 1982. Audience design in meaning and reference. In J. F. LeNy and W. Kintsch, editors, *Language and comprehension*. Amsterdam: North-Holland.

N. Dethlefs and H. Cuayahuitl. 2010. Hierarchical Reinforcement Learning for Adaptive Text Generation. In *Proc. INLG 2010*.

K. Georgila, J. Henderson, and O. Lemon. 2005. Learning User Simulations for Information State Update Dialogue Systems. In *Proc of Eurospeech/Interspeech*.

F. Hernandez, E. Gaudioso, and J. G. Boticario. 2003. A Multiagent Approach to Obtain Open and Flexible User Models in Adaptive Learning Communities. In *User Modeling 2003*, volume 2702/2003 of *LNCS*. Springer, Berlin / Heidelberg.

E. A. Issacs and H. H. Clark. 1987. References in conversations between experts and novices. *Journal of Experimental Psychology: General*, 116:26–37.

S. Janarthanam and O. Lemon. 2009a. A Two-tier User Simulation Model for Reinforcement Learning of Adaptive Referring Expression Generation Policies. In *Proc. SigDial'09*.

S. Janarthanam and O. Lemon. 2009b. A Wizard-of-Oz environment to study Referring Expression Generation in a Situated Spoken Dialogue Task. In *Proc. ENLG'09*.

S. Janarthanam and O. Lemon. 2009c. Learning Lexical Alignment Policies for Generating Referring Expressions for Spoken Dialogue Systems. In *Proc. ENLG'09*.

S. Janarthanam and O. Lemon. 2010. Learning to Adapt to Unknown Users: Referring Expression Generation in Spoken Dialogue Systems. In *Proc. ACL'10*.

O. Lemon, Georgila. K., and J. Henderson. 2006. Evaluating Effectiveness and Portability of Reinforcement Learned Dialogue Strategies with real users: the TALK TownInfo Evaluation. In *IEEE/ACL Spoken Language Technology*.

O. Lemon. 2010. Learning what to say and how to say it: joint optimization of spoken dialogue management and Natural Language Generation. *Computer Speech and Language*. (to appear).

E. Levin, R. Pieraccini, and W. Eckert. 1997. Learning Dialogue Strategies within the Markov Decision Process Framework. In *Proc. of ASRU97*.

V. Rieser and O. Lemon. 2009. Natural Language Generation as Planning Under Uncertainty for Spoken Dialogue Systems. In *Proc. EACL'09*.

V. Rieser and O. Lemon. 2010. Optimising information presentation for spoken dialogue systems. In *Proc. ACL*. (to appear).

J. Schatzmann, K. Weilhammer, M. N. Stuttle, and S. J. Young. 2006. A Survey of Statistical User Simulation Techniques for Reinforcement Learning of Dialogue Management Strategies. *Knowledge Engineering Review*, pages 97–126.

J. Schatzmann, B. Thomson, K. Weilhammer, H. Ye, and S. J. Young. 2007. Agenda-based User Simulation for Bootstrapping a POMDP Dialogue System. In *Proc of HLT/NAACL 2007*.

D. Shapiro and P. Langley. 2002. Separating skills from preference: Using learning to program by reward. In *Proc. ICML-02*.

R. Sutton and A. Barto. 1998. *Reinforcement Learning*. MIT Press.