CLASSiC

D3.4: Agenda-based user simulations and EM training tools for Appointment Scheduling and TownInfo domains

Simon Keizer, Olivier Pietquin, Stéphane Rossignol, and Steve Young

Distribution: Public

CLASSiC
Computational Learning in Adaptive Systems for Spoken Conversation
216594 Deliverable 3.4

October 2010

Project funded by the European Community
under the Seventh Framework Programme for
Research and Technological Development

The deliverable identification sheet is to be found on the reverse of this page.

Project ref. no.	216594
Project acronym	CLASSiC
Project full title	Computational Learning in Adaptive Systems for Spoken Conversation
Instrument	STREP
Thematic Priority	Cognitive Systems, Interaction, and Robotics
Start date / duration	01 March 2008 / 36 Months

Security	Public
Contractual date of delivery	M30 = August 2010
Actual date of delivery	October 2010
Deliverable number	3.4
Deliverable title	D3.4: Agenda-based user simulations and EM training tools for Appointment Scheduling and TownInfo domains
Type	Prototype
Status & version	Draft 1.0
Number of pages	18 (excluding front matter)
Contributing WP	3
WP/Task responsible	UCAM, SUPELEC
Other contributors	
Author(s)	Simon Keizer, Olivier Pietquin, Stéphane Rossignol, and Steve Young
EC Project Officer	Philippe Gelin
Keywords	Spoken dialogue systems, dialogue management, user simulation

The partners in CLASSiC are:

Heriot-Watt University	HWU
University of Cambridge	UCAM
University of Geneva	GENE
Ecole Superieure d'Electricite	SUPELEC
France Telecom/ Orange Labs	FT
University of Edinburgh HCRC	EDIN

For copies of reports, updates on project activities and other CLASSiC-related information, contact:

The CLASSiC Project Co-ordinator:
Dr. Oliver Lemon
School of Mathematical and Computer Sciences (MACS)
Heriot-Watt University
Edinburgh
EH14 4AS
United Kingdom
O.Lemon@hw.ac.uk
Phone +44 (131) 451 3782 - Fax +44 (0)131 451 3327

Copies of reports and other material can also be accessed via the project's administration homepage, `http://www.classic-project.org`

Contents

Executive summary

This document is a short report to accompany the Prototype deliverable 3.4, due at month 30 of the CLASSiC project. The contribution consists of two parts: 1) the agenda-based user simulator and associated parameter estimation tools, and 2) the dynamic Bayesian network simulator and associated EM training tools. Both simulation approaches are applicable in the TownInfo and Appointment Scheduling domains. 4 project publications relate directly to this deliverable, and their abstracts are presented in the Appendix. They are available at `www.classic-project.org`.

Chapter 1

Introduction

This document describes the prototype deliverable consisting of the agenda-based user simulator developed by Cambridge University, which will be described in Chapter 2, and the probabilistic user simulator developed by SUPELEC, which will be described in Chapter 3. The agenda-based simulator incorporates random decision points controlled by probability distributions that can be estimated from corpus data and is used for training and evaluating dialogue management policies of the Cambridge POMDP dialogue managers. The probabilistic user simulator uses a dynamic Bayesian network for generating user actions and its parameters can be trained from data using EM techniques. Both simulators can be used for the TownInfo and Appointment Scheduling domains.

Chapter 2

Agenda based user simulation

2.1 Introduction

In order to test, evaluate, and train POMDP dialogue managers, a user simulator has been developed, which interacts with the dialogue manager on the semantic level [1]. This means that it can process incoming system dialogue acts and return user response dialogue acts. An additional error model is used to simulate speech understanding errors, transforming the correct simulated user act into an n-best list of (possibly) confused dialogue acts.

All task-domains supported by the Cambridge POMDP dialogue systems and the user simulator are defined through an ontology and a database. The ontology specifies the valid (combinations of) slots and their values, whereas the database contains entities specified in terms of these slots and values. In any dialogue between the simulated user and the system, the user tries to find an entity according to his preferences and the system tries to find out these preferences and accesses the database to provide information about an appropriate entity. In the case of the TownInfo domain, the entities are touristic venues in a town and the slots describe features such as area, price range, phone number, and address. In the Appointment Scheduling domain, the entities are free slots on the system's calendar, described in terms of date, the day of the week, the month, etcetera.

The main components of the agenda-based user simulator are the *user goal* and the *agenda*. The user goal consists of a list of constraints in the form of slot-value pairs, describing the user's preferences regarding the entity to be found. The agenda contains user dialogue acts that are planned responses to the system. In each turn, the simulator updates his goal and agenda based on the system act and generates a response act by popping items from the agenda. This allows for modelling user behaviour that provides the complexity and rationality to be able to train and evaluate dialogue management policies. For example, the user can correct misunderstandings of the system or change his mind by relaxing or changing one of their constraints.

2.2 TownInfo example

Below is an example dialogue between the simulated user and the HIS dialogue manager [2], in which the user has the goal to get the name, phone number and address of a bar that plays ethnic music and serves wine. The dialogue shows both system (Sys) and user (Usr) acts, as well as the n-best list of user act

hypotheses (Rec) with confidence scores (between curly brackets) generated by the error model. At the start of the dialogue, the simulator is initialised with a (randomly generated) user goal with constraints (G_C) and requested slots (G_R). At the end, a dialogue score is assigned, based on whether the dialogue was successful and the number of turns.

```
G_R> name, phone, addr
G_C> type=bar, music=Ethnic, drinks=wine

Sys> hello() [Greet]
Usr> hello() {1.00}
Rec> hello() {0.32}
     thankyou() {0.32}
     ack() {0.32}
     .
Sys> repeat() [Repeat]
Usr> hello(=bar) {1.00}
Rec> reqmore() {0.32}
     hello(=bar) {0.32}
     repeat() {0.32}
     .
Sys> request(near) [BoldRQ]
Usr> inform(music=Ethnic,drinks=wine) {1.00}
Rec> deny() {0.32}
     inform(music=Ethnic,drinks=wine) {0.32}
     reqmore() {0.32}
     .
Sys> inform(name=none,type=bar,music=Ethnic) [Offer]
Usr> reqalts(music=Pop) {1.00}
Rec> reqalts(music=Pop) {1.00}
     .
Sys> inform(name="Carnaby Street",type=bar,drinks=beer,music=Pop) [Offer]
Usr> deny(drinks=beer,drinks=wine) {1.00}
Rec> deny(drinks=cocktails,drinks=wine) {0.32}
     deny(drinks=beer,drinks=wine) {0.32}
     request(stars) {0.32}
     .
Sys> inform(name="The Parrot",type=bar,drinks=wine,music=Pop) [Offer]
Usr> request(phone) {1.00}
Rec> request(phone) {0.93}
     ack() {0.06}
     .
Sys> inform(name="The Parrot",phone="728-66-32") [Inform]
Usr> request(addr) {1.00}
Rec> request(addr) {0.93}
     request(pricerange) {0.06}
     .
Sys> inform(name="The Parrot",addr="Art Square") [Inform]
Usr> bye()
Rec> hello() {0.32}
     confirm(drinks=dontcare) {0.32}
     bye() {0.32}
     .

End> .

Dialogue score 13.00
```

2.3 Appointment Scheduling example

For the Appointment Scheduling domain, the simulator uses a (randomly generated) user calendar, containing the availability of the user on each date/time slot in the two weeks following a given start date. The user tries to find a free slot which is also available for the system by generating goals from the calendar representing either an individual free date/time slot or an interval of free date/time slots. For each generated goal, the user tries to find out if the system can offer a matching free slot, which is similar to finding a venue matching a set of constraints in the tourist information domain. If a goal cannot be satisfied, the corresponding date/time slots are marked as 'rejected' on the calendar and a new goal is generated, and so on until a matching time slot has been found and an appointment can be made.

Below is an example of a dialogue between the simulator and the initial version of the BUDS Appointment Scheduling dialogue manager, for which a POMDP policy will be trained using the simulator. The initial user calendar is given, in which the first free slot the user wants to make an appointment on (January 9, PM), as well as the eventual appointment with the system (January 18, PM) are underlined.

Two-week calendar from reference date 2010-Jan-08 (= day 1 = 'today'):

date	day	week	status (AM)	status (PM)
2010-Jan-09	Saturday	1	busy	**free**
2010-Jan-10	Sunday	1	busy	free
2010-Jan-11	Monday	2	free	free
2010-Jan-12	Tuesday	2	free	busy
2010-Jan-13	Wednesday	2	busy	busy
2010-Jan-14	Thursday	2	free	free
2010-Jan-15	Friday	2	busy	free
2010-Jan-16	Saturday	2	free	free
2010-Jan-17	Sunday	2	free	free
2010-Jan-18	Monday	3	free	**free**
2010-Jan-19	Tuesday	3	free	free
2010-Jan-20	Wednesday	3	free	free
2010-Jan-21	Thursday	3	free	busy

```
First available slot:
Day 2; 2010-Jan-09 Saturday (pm): free [week 1]

G_C> week=1, dayofmonth=9, dayofweek=Saturday, time=pm, month=January

Sys> hello()
Usr> hello() {1.00}
     .
Sys> request(timetype)
Usr> inform(time=pm) {1.00}
     .
Sys> request(dayofweektype)
Usr> inform(dayofweek=Saturday) {1.00}
     .
Sys> inform(name=none,dayofweek=Saturday,time=pm)
Usr> inform(dayofweek=Sunday) {1.00}
     .
Sys> inform(name=none,dayofweek=Sunday,time=pm)
Usr> inform(time=am) {1.00}
     .
Sys> inform(name=none,time=am,dayofweek=Sunday)
Usr> negate(dayofweek=Monday) {1.00}
     .
Sys> inform(name=none,dayofweek=Monday,time=am)
Usr> inform(time=pm) {1.00}
     .
Sys> inform(name="2010-01-18 pm",time=pm,dayofweek=Monday,booking=option)
Usr> ack() {1.00}
```

```
     .
Sys> inform(name="2010-01-18 pm",booking=final)
Usr> bye() {1.00}
     .

End> .

Dialogue score 11.00
```

2.4 Parameter estimation

The original agenda-based user simulator was extended to incorporate so-called *random decision points*. During the processing of a system act, these can be encountered depending on the context and when encountered, a random decision is made between several options by sampling from a multinomial or geometric probability distribution. For example, when the system offered a venue, the user randomly decides to ask for an alternative and/or change his goal.

A sample-based, maximum likelihood method was developed to estimate the parameters for the random decision distributions from data [3]. For the TownInfo domain, the March 2009 corpus as described in D6.3 [4] was used to estimate the parameters, using the semantic annotations of the transcribed user utterances. Since the parameter estimation method is domain-independent, parameters can also be estimated for the Appointment Scheduling domain, provided that suitable corpus data is available in which the user behaviour displays the variation that is captured by the random decision points.

2.5 Software

The agenda-based user simulation and associated parameter estimation software is structured as follows:

- **atk**: general application toolkit
- ***SemIO***: semantic decoder software used for word-level error model;
- **tHIS**: C++ sources of dialogue management and user simulation software, including:
  - *BUDSLib*: the BUDS dialogue manager;
  - *HISLib*: the HIS dialogue manager;
  - ***UMLib***: user simulation error modelling library;
    - **UserModelTraining**: C++ program for estimating the user simulation parameters from corpus data
    - **ErrorModelTraining**: C++ program for estimating semantic level confusion model statistics for the error model
  - *TDMan*: the test harness to run the simulator with one of the dialogue managers;
- ***resources***: the domain ontology and database, configuration files, parameter files for user simulator and error model;

In the README files included in the package, more specific documentation is given on how to compile the system and how to run the programs for training and evaluating the parameters and for training a semantic level statistical error model.

The UMLib library itself can be used to run the simulator and error model with a dialogue manager. The class `UM` provides the main interface to the library. To interact with it, following three public functions are used:

- `void UM::Init(void)` : initialises the user simulator
- `void UM::Receive(DiaActPtr a_m)` : is called to transmit a system act a_m to the user model
- `DiaActPtr UM::Respond(void)` : is called to generate the user response

The DManUtils library contains a `GoalGenerator` class that is used to generate random goals (class `UMGoal`) the user simulator is initialised with in each dialogue.

Finally, the CorpusLib library contains classes to evaluate dialogues on the basis of either 1) the system and user acts and the predefined goal (class `EvalUtils`) or 2) the system and user acts only (class `TaskScorer`), where the user goal is inferred.

Chapter 3

EM training tools

In this chapter, the training methods used for estimating the parameters of the Bayesian-network-based (BN-based) user simulation developed by SUPELEC are described [5, 6, 7].

In the case of man-machine dialogue data, some information is often missing in the annotations. For instance, the user's internal representation of the dialogue context is unknown. In the remainder of this chapter and related publications, this representation will be referred to as the *knowledge* of the user.

The knowledge of the user can be inferred from the data itself, by a human expert, a set of rules, or a trained classification algorithm dedicated to this task. In Section 3.1, the latter approach is followed, and the derived training methods for learning the BN parameters are explained. Alternatively, the knowledge of the user can treated as hidden and the BN parameters can be learned using the Expectation-Maximisation algorithm. This approach is described in Section 3.2, both within a statistical framework (expected-likelihood maximisation) and within a Bayesian framework (starting from some prior distribution over parameters).

Evaluation results for both approaches are given in Section 3.3.

3.1 Training methods without missing data

3.1.1 Maximum likelihood

When all variables in a dataset are observed, a statistical framework can be used, in which the frequency of events appearing in the database are computed. This is known as the *maximum likelihood* approach:

$$\Theta_{i,j,k}^{ML} = \hat{p}\left(X_i = x_k \mid pa(X_i) = x_j\right) = \frac{N_{i,j,k}}{\sum_k N_{i,j,k}}$$

where the set of $\Theta_{i,j,k}^{ML}$ are the BN parameters that need to be learned, $N_{i,j,k}$ is the number of events in the database for which the variable X_i is in the state x_k and its parents in the network (pa) in the configuration x_j.

3.1.2 Bayesian training

Bayesian estimation of the parameters is slightly different. It actually aims at estimating the probability distribution over parameters and estimates the parameters using either a *maximum a posteriori* (MAP) approach or the parameters' expectation given this distribution. This is done knowing that the variables have been observed and requires some prior on the parameters. Using a Dirichlet distribution prior (standard choice for multivariate distributions), it is possible to derive an analytical formula for the expected parameters which is similar to the one obtained in the previous section. Using the MAP approach:

$$\Theta_{i,j,k}^{MAP} = \hat{p}\left(X_i = x_k \mid pa(X_i) = x_j\right) = \frac{N_{i,j,k} + \alpha_{i,j,k} - 1}{\sum_k N_{i,j,k} + \alpha_{i,j,k} - 1}$$

where the $\alpha_{i,j,k}$ are the coefficients of the Dirichlet distribution.
Using the *a priori* expectation approach (AEP) instead of the MAP, one gets:

$$\Theta_{i,j,k}^{EAP} = \hat{p}\left(X_i = x_k \mid pa(X_i) = x_j\right) = \frac{N_{i,j,k} + \alpha_{i,j,k}}{\sum_k N_{i,j,k} + \alpha_{i,j,k}}$$

3.1.3 Priors on parameters

The $\alpha_{i,j,k}$ are Dirichlet distribution coefficients and define priors on the distribution parameters, as they are set by an expert. It is thus possible to give to these coefficients more or less importance, given the confidence of the expert. This will result in different trained BN/retrained BN user simulators. Fine-tuning the $\alpha_{i,j,k}$ will allow us to get simulators behaving more or less like the human users which produced the database, as shown in section 3.3. Of course, if nothing is known (no expert available), a uniform distribution over parameters (all coefficient being equal) can be taken as a prior and the method can still be used.

3.2 Training methods with missing data

3.2.1 Expectation-Maximisation algorithm

The *Expectation-Maximisation* (EM) algorithm allows estimating the BN parameters even when the data corresponding to some of the parameters is missing.
EM is a recursive algorithm applied until convergence as explained hereafter.
Let us assume that:

- $X_{\mathrm{v}} = \left\{X_{\mathrm{v}}^{(l)}\right\}_{l=1\ldots N}$ is the set of the N observable data.
- $\Theta^{(t)} = \left\{\Theta_{i,j,k}^{(t)}\right\}$ are the estimations of the parameters of the BN at iteration t.

EM is a recursive algorithm, initialised with arbitrary $\Theta^{(0)}$ values, consisting of two steps:

- **Expectation (E)** step: the missing data $N_{i,j,k}$ are estimated, by computing their expectation conditionally to the data and to the current parameter estimates (i.e., to the current distribution estimate):

$$N^*_{i,j,k} = E[N_{i,j,k}] = \sum_{l=1}^{N} \hat{p}\left(X_i = x_k \mid pa(X_i) = x_j, X_v^{(l)}, \Theta^{(t)}\right)$$

This consists in doing inference using the current parameter values, and in replacing the missing values by the probabilities obtained by inference.

- **Maximisation (M)** step: replacing the missing $N_{i,j,k}$ by their expected value computed in the previous step, it is possible to compute the new parameter values $\Theta^{(t+1)}$, using maximum likelihood:

$$\Theta^{(t+1)}_{i,j,k} = \frac{N^*_{i,j,k}}{\sum_k N^*_{i,j,k}}$$

3.2.2 Expectation-Maximisation algorithm and Bayesian training

The EM algorithm can be used within the Bayesian framework as well. In that case, the maximum likelihood estimation used in the **M** step must be replaced by an *a posteriori* maximum. Using the *a posteriori* expectation, one gets:

$$\Theta^{(EM)} = \Theta^{(t+1)}_{i,j,k} = \frac{N^*_{i,j,k} + \alpha_{i,j,k}}{\sum_k N^*_{i,j,k} + \alpha_{i,j,k}}$$

3.3 Results

The BN-based user simulator has been tested against the UCAM Dialogue Manager. Six configurations for the BN-based user simulator were tested. 1000 dialogues were generated for each configuration. The six configurations are described below:

- "ori-T-BN": the knowledge parameters were estimated on the database and the BN parameters were learned using the results by a Maximum Likelihood method ($\Theta^{ML}_{i,j,k}$) (see Section 3.1).
- "mod-T-BN": the knowledge parameters were estimated from the database and the BN parameters ($\Theta^{EAP}_{i,j,k}$) were learned with a Bayesian learning method (EAP method) and using priors fixed by an expert, given moderate importance (see Section 3.1.3).
- "H-BN": the BN parameters were hand-coded by an expert (Heuristics).
- "mod-T1-BN": the knowledge was supposed missing and the BN parameters ($\Theta^{(EM)}$) were learned using the database by Bayesian EM and priors fixed by an expert; first version: the expert knowledge is given minimal importance (see Section 3.1.3).
- "mod-T2-BN": the knowledge was supposed missing and the BN parameters ($\Theta^{(EM)}$) were learned using the database by Bayesian EM and priors fixed by an expert; second version: the expert knowledge is given moderate importance (see Section 3.1.3).

	ori-T-BN	mod-T-BN	H-BN
Precision:	47.11	50.62	63.63
Recall:	57.89	60.68	53.20
KL:	0.7292	0.6712	0.8803
Nturns/diag:	18.19	15.15	5.283

Table 3.1: Dissimilarities using the first three BN configurations

	mod-T1-BN	mod-T2-BN	mod-T3-BN
Precision:	63.71	64.60	67.13
Recall:	61.84	63.83	69.27
KL:	0.6674	0.7864	0.5288
Nturns/diag:	7.690	7.980	8.703

Table 3.2: Dissimilarities using the last three BN configurations

- "mod-T3-BN": the knowledge was supposed missing and the BN parameters ($\Theta^{(EM)}$) were learned using the database by Bayesian EM and priors fixed by an expert; third version: the expert knowledge is given high importance.

The last three configurations are the most realistic ones.

Four dissimilarity measures (see D3.5) have been computed: the Precision, the Recall, the Kullback-Leibler (KL) and the mean number of turns per dialog. The simulated dialogues are compared to the dialogues from the database on this basis. Notice that the Precision and the Recall must be as high as possible, the Kullback-Leibler as low as possible and the mean number of turns per dialogue as close to the mean number of turns per dialogue in the dialogues from the database (which is 8.185). The results are given in Tables 3.1 and 3.2.

Table 3.1 clearly indicates that the first configurations do not provide realistic dialogues. Considering the Recall, the KL and the number of turns, the mod-T-BN gives the best results. The fact that ori-T-BN gives bad results indicates that the database is not large enough, and/or that the inferred knowledge is not very accurate. The H-BN was designed to give as short as possible dialogues: this can be seen in the dissimilarity measures.

Table 3.2 indicates that the training techniques with missing data are efficient, allowing us not to use the error-prone knowledge inference. Taking the expert information into account allows to improve the performance to some extent, considering the Precision, the Recall and the number of turns per dialogue dissimilarity measures. The KL dissimilarity measure gives more uncertain results.

Chapter 4

Conclusions

This document has described the prototype deliverable consisting of two user simulator approaches for both the TownInfo and Appointment Scheduling domains. The agenda-based user simulator developed by Cambridge University incorporates random decision points controlled by probability distributions that can be estimated from corpus data and is used for training and evaluating dialogue management policies of the Cambridge POMDP dialogue managers. The probabilistic user simulator developed by SUPELEC uses a dynamic Bayesian network for generating user actions and its parameters can be trained from data using EM techniques.

Bibliography

[1] J. Schatzmann, B. Thomson, K. Weilhammer, H. Ye, and S. Young. Agenda-based user simulation for bootstrapping a POMDP dialogue system. In *Proceedings HLT/NAACL*, Rochester, NY, 2007.

[2] S. Young, M. Gašić, S. Keizer, F. Mairesse, B. Thomson, and K. Yu. The Hidden Information State model: a practical framework for POMDP based spoken dialogue management. *Computer Speech and Language*, 24(2):150–174, April 2010.

[3] S. Keizer, M. Gašić, F. Jurčíček, F. Mairesse, B. Thomson, K. Yu, and Steve Young. Parameter estimation for agenda-based user simulation. In *Proc. SIGdial*, Tokyo, Japan, September 2010.

[4] P. Bretier, P. Crook, S. Keizer, R. Laroche, O. Lemon, and G. Putois. Initial evaluation of towninfo and self-help systems. Technical Report D6.3, CLASSiC, June 2009.

[5] Olivier Pietquin and Thierry Dutoit. A probabilistic framework for dialog simulation and optimal strategy learning. *IEEE Transactions on Audio, Speech and Language Processing*, 14(2):589–599, mar 2006.

[6] Olivier Pietquin, Stéphane Rossignol, and Michel Ianotto. Training bayesian networks for realistic man-machine spoken dialogue simulation. In *Proceedings of the 1rst International Workshop on Spoken Dialogue Systems Technology (IWSDS 2009)*, Irsee (Germany), December 2009. 4 pages.

[7] Stéphane Rossignol, Olivier Pietquin, and Michel Ianotto. Grounding simulation in spoken dialog systems with bayesian networks. In G. Geunbae Lee et al., editor, *Proceedings of the International Workshop on Spoken Dialogue Systems (IWSDS 2010)*, volume 6392 of *Lecture Notes in Artificial Intelligence (LNAI)*, pages 110–121, Gotemba (Japan), October 2010. Springer-Verlag, Heidelberg-Berlin.

Appendix A

Abstracts of publications associated with this deliverable

Parameter estimation for agenda-based user simulation

S. Keizer, M. Gašić, F. Jurčíček, F. Mairesse, B. Thomson, K. Yu, and S. Young

In *Proceedings SIGdial*, Tokyo, Japan, 2009.

Abstract:

This paper presents an agenda-based user simulator which has been extended to be trainable on real data with the aim of more closely modelling the complex rational behaviour exhibited by real users. The trainable part is formed by a set of *random decision points* that may be encountered during the process of receiving a system act and responding with a user act. A sample-based method is presented for using real user data to estimate the parameters that control these decisions. Evaluation results are given both in terms of statistics of generated user behaviour and the quality of policies trained with different simulators. Compared to a handcrafted simulator, the trained system provides a much better fit to corpus data and evaluations suggest that this better fit should result in improved dialogue performance.

The Hidden Information State model: a practical framework for POMDP based spoken dialogue management

S. Young, M. Gašić, S. Keizer, F. Mairesse, B. Thomson, and K. Yu

In *Computer Speech and Language*, 24(2):150-174, April 2010.

Abstract:

This paper explains how Partially Observable Markov Decision Processes (POMDPs) can provide a principled mathematical framework for modelling the inherent uncertainty in spoken dialogue systems. It briefly summarises the basic mathematics and explains why exact optimisation is intractable. It then describes in some detail a form of approximation called the *Hidden Information State model* which does scale and which can be used to build practical systems. A prototype HIS system for the tourist information domain is evaluated and compared with a baseline MDP system using both user simulations and a live user trial. The results give strong support to the central contention that the POMDP-based framework is both a tractable and powerful approach to building more robust spoken dialogue systems.

Training Bayesian networks for realistic man-machine spoken dialogue simulation

O. Pietquin, S. Rossignol, M. Ianotto

In *Proceedings of the 1st International Workshop on Spoken Dialogue Systems Technology (IWSDS 2009)*, 4 pages, December 2009.

Abstract:

Data collection and annotation are generally required to design or assess spoken dialogue systems. Yet, this is a very time consuming and expensive process. For these reasons, user simulation has become an important trend of research in the field of spoken dialogue systems. The general problem of user simulation is thus to produce as many as necessary natural, various and consistent interactions from as few data as possible. In this paper, we propose a user simulation method based on Bayesian networks (BN) that is able to produce consistent interactions in terms of user goal and dialogue history. The model as been introduced in previous work but parameters were hand-tuned and it was assessed in the framework of automatic learning of optimal dialogue strategies. In this paper, the BN is trained on a database of 1234 human-machine dialogues in the TownInfo domain (a tourist information application). Experiments with a state-of-the-art dialogue system (REALL-DUDE/DIPPER/OAA) have been realised and results in terms of dialog statistics are presented.

Grounding Simulation in Spoken Dialog Systems with Bayesian Networks

S. Rossignol, O. Pietquin, M. Ianotto

In *Proceedings of the 2nd International Workshop on Spoken Dialogue Systems Technology (IWSDS 2010)*, G. Geunbae Lee et al., Springer-Verlag, Heidelberg-Berlin, Lecture Notes in Artificial Intelligence (LNAI), 6392:110-121, Gotemba (Japan), 2010

Abstract:

User simulation has become an important trend of research in the field of spoken dialog systems because collecting and annotating real man-machine interactions with users is often expensive and time consuming. Yet, such data are generally required for designing and assessing efficient dialog systems. The general problem of user simulation is thus to produce as many as necessary natural, various and consistent interactions from as few data as possible. In this paper, is proposed a user simulation method based on *Bayesian Networks* (BN) that is able to produce consistent interactions in terms of user goal and dialog history but also to simulate the grounding process that often appears in human-human interactions. The BN is trained on a database of 1234 human-machine dialogs in the TownInfo domain (a tourist information application). Experiments with a state-of-the-art dialog system (REALL-DUDE/DIPPER/OAA) have been realised and promising results are presented.