Chapter from the book *Machine Learning*
Downloaded from: http://www.intechopen.com/books/machine_learning

8

Machine Learning Methods for Spoken Dialogue Simulation and Optimization

Olivier Pietquin
Ecole Supérieure d'Electricité (Supélec)
France

1. Introduction

Computers and electronic devices are becoming more and more present in our day-to-day life. This can of course be partly explained by their ability to ease the achievement of complex and boring tasks, the important decrease of prices or the new entertainment styles they offer. Yet, this real incursion in everybody's life would not have been possible without an important improvement of Human-Computer Interfaces (HCI). This is why HCI are now widely studied and become a major trend of research among the scientific community. Designing "user-friendly" interfaces usually requires multidisciplinary skills in fields such as computer science, ergonomics, psychology, signal processing etc. In this chapter, we argue that machine learning methods can help in designing efficient speech-based human-computer interfaces.

Speech is often considered as the most convenient and natural way for humans to communicate and interact with each other. For this reason, speech and natural language processing have been intensively studied for more than 60 years. It has now reached a maturity level that should enable the design of efficient voice-based interfaces such as Spoken Dialogue Systems (SDS). Still, designing and optimizing a SDS is not only a matter of putting together speech and language processing systems such as Automatic Speech Recognition (ASR) (Rabiner & Juang 1993), Spoken Language Understanding (SLU) (Allen 1998), Natural Language Generation (NLG) (Reiter & Dale 2000), and Text-to-Speech (TTS) synthesis (Dutoit 1997) systems. It also requires the development of dialogue strategies taking at least into account the performances of these subsystems (and others), the nature of the task (e.g. form filling (Pietquin & Dutoit 2006a), tutoring (Graesser *et al* 2001), robot control, or database querying (Pietquin 2006b)), and the user's behaviour (e.g. cooperativeness, expertise (Pietquin 2004)). The great variability of these factors makes rapid design of dialogue strategies and reusability across tasks of previous work very complex. For these reasons, human experts are generally in charge of tailoring dialogue strategies which is costly and time-consuming. In addition, there is also no objective way to compare strategies designed by different experts or to objectively qualify their performance. Like for most software engineering tasks, such a design is a cyclic process. Strategy hand-coding, prototype releases and user tests are required making this process expensive and time-consuming.

In the purpose of obtaining automatic data-driven methods and objective performances measures for SDS strategy optimisation, statistical learning of optimal dialogue strategies

Source: Machine Learning, Book edited by: Abdelhamid Mellouk and Abdennacer Chebira,
ISBN 978-3-902613-56-1, pp. 450, February 2009, I-Tech, Vienna, Austria

became a leading domain of research (Lemon & Pietquin, 2007). The goal of such approaches is to reduce the number of design cycles (Fig.1).

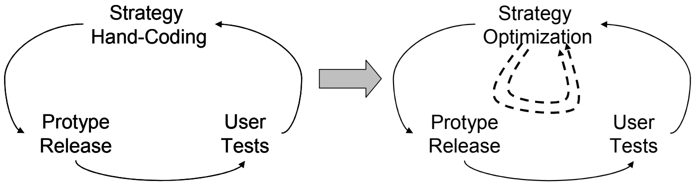


Fig. 1.Optimization for minimizing the number of design cycles

Supervised learning for such an optimization problem would require examples of ideal (sub)strategies which are typically unknown. Indeed, no one can actually provide an example of what would have objectively been the perfect sequencing of exchanges after having participated to a dialogue. Humans have a greater propensity to criticize what is wrong than to provide positive proposals. In this context, reinforcement learning using Markov Decision Processes (MDPs) (Levin *et al* 1998, Singh *et al* 1999, Scheffler & Young 2001, Pietquin & Dutoit 2006a, Frampton & Lemon 2006) and Partially Observable MDP (POMDPs) (Poupart *et al* 2005, Young 2006) has become a particular focus.

Such machine learning methods are very data demanding and sufficient amounts of annotated dialogue data are often not available for training. Different standard methods have therefore been investigated to deal with the data sparsity that can be split into two classes: statistical generation of new data by means of simulation (Schatzmann *et al*, 2007a) or generalization to unseen situations (Henderson *et al*, 2005).

In this chapter, we propose to provide an overview of the state of the art in machine learning for spoken dialogue systems optimization. This will be illustrated on a simple train ticket booking application.

2. Definitions and formalisms

2.1 Definitions

In this text, a *dialogue* will be describing an interaction between two *agents* based on sequential turn taking. We will only treat the special case of *goal-directed* dialogs where both agents cooperate in order to achieve an aim (or accomplish a task), like obtaining a train ticket for example. *Social* dialogues are out of the scope of this chapter. We will consider *man-machine* dialogs where one of the agents is a human *user* while the other is a computer (or *system*). In the particular case of a speech-based communication, the computer implements a *Spoken Dialogue System* (SDS). When one of the agents is an SDS, the dialogue consists of a sequence of *utterances* exchanged at each turn. A *spoken utterance* is the acoustic realisation of the *intentions* or *concepts* (or *dialog acts*, *communicative acts*) one of the agents wants to communicate to the other and is expressed as a *word* sequence. The amount of time between one communication and the other can be of variable length and is called a *turn*.

2.2 Formal description of man-machine spoken dialog

So as to use statistical machine learning for SDS strategy optimization, one needs to describe a spoken dialogue in terms of a finite number of variables. A man-machine spoken dialog will therefore be considered as a sequential (turn-taking) process in which a human user and a Dialog Manager (DM) communicate using spoken utterances passing through speech and language processing modules (Fig.2). A Knowledge Base (KB) is usually connected to the DM and contains information about the task addressed by the system (i.e. a list of songs).

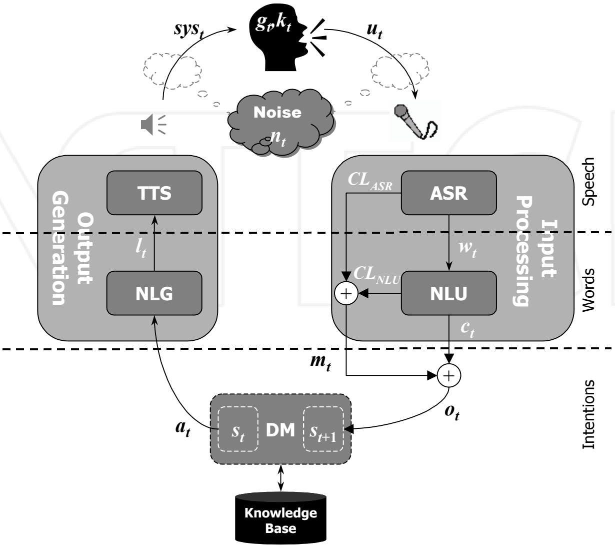


Fig. 2. Man-Machine Spoken Communication

The DM implements the SDS strategy or *policy* π defining a mapping between the DM internal state and dialogue acts (the way to build the DM internal state will be discussed later). It thus takes decisions about what to *say* at a given time. Dialogue acts can be of different kinds: providing information to the user, asking for more information, closing the dialogue, etc. The DM decisions (so its policy) are of course of a major importance since they make the interaction going in one direction or another. Adopting the system's point of view, the information exchange typically starts at turn t with the generation of a communicative act $\boldsymbol{a_t}$ by the DM. This act is generated according to the DM's strategy $\boldsymbol{\pi_t}$ and internal state $\boldsymbol{s_t}$ at turn t, and has to be transformed in a spoken output. A Natural Language Generation (NLG) module converts this act into a linguistic representation $\boldsymbol{l_t}$ (generally a text) which in turn serves as an input to a Text-to-Speech (TTS) system. The output of the TTS module is a spoken utterance $\boldsymbol{sys_t}$ addressed to the user. From this, the human user produces a new spoken utterance $\boldsymbol{u_t}$ taking into account his/her understanding of sys_t but also to his/her background knowledge $\boldsymbol{k_t}$ (about the task, the interaction history, the world in general) and finally to the goal $\boldsymbol{g_t}$ s/he is pursuing while interacting with the system. Both utterances sys_t and u_t can be mixed with some additional environmental noise $\boldsymbol{n_t}$. This potentially noisy user utterance is then processed by an ASR system which output is a sequence of words $\boldsymbol{w_t}$ as well as a confidence level $\boldsymbol{CL_{ASR}}$ associated to this result. The sequence $\boldsymbol{w_t}$ is usually taken out of a so called "Nbest list" ranking the best hypotheses the system can make about what the user said given the speech signal. The confidence level is usually a number between 0 and 1 providing information about the confidence the systems in the result of its processing.

It can also be a real number, depending on the system. Finally, the Natural Language Understanding (NLU) module generates a set of concepts (or communicative acts) $\boldsymbol{c_t}$ also picked from a "Nbest list" derived from $\boldsymbol{w_t}$ and again with a confidence level $\boldsymbol{CL_{NLU}}$. The observation $\boldsymbol{o_t}$ passed to the DM is actually the set $\{c_t, CL_{ASR}, CL_{NLU}\}$. From this observation, a new internal state is computed by the DM which will be used to generate a new dialog act $\boldsymbol{a_{t+1}}$. A new cycle is then started again until the end of the dialogue. This can occur when the user reached his/her goal or whenever the user or the system wants to stop the interaction for any reason (dissatisfaction, looping dialogue etc.)

2.3 Reinforcement learning and Markov decision processes

From the former description of a spoken dialogue system, it is clear that optimizing a SDS is about implementing an optimal strategy into the dialogue manager. Adopting a machine learning point of view, automatic optimization of a strategy is addressed by Reinforcement Learning (RL). The general purpose of a RL agent is to optimally control a stochastic dynamic system. The control problem is then described in terms of states, actions and rewards. In this framework, an artificial agent tries to learn an optimal control policy through real interactions with the system. It observes the state s of the system through an observation o and chooses an action a to apply on it accordingly to a current internal policy π mapping states to actions. A feedback signal r is provided to the agent after each interaction as a reward information, which is a local hint about the quality of the control. This reward is used by the agent to incrementally learn the optimal policy, simply by maximizing a function of the cumulative rewards.

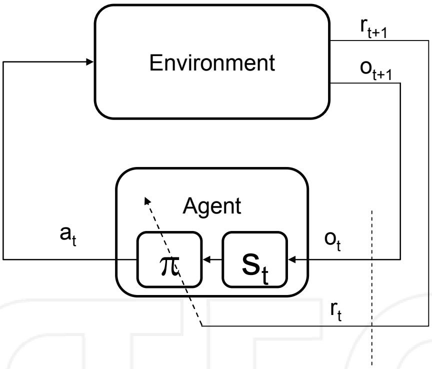


Fig. 3. Reinforcement Learning paradigm

This can be put into the formalism of Markov Decision Processes (MDP), where a discrete-time system interacting with its stochastic environment through actions is described by a finite or infinite number of states $\{s_i\}$ in which a given number of actions $\{a_j\}$ can be performed. To each state-action pair is associated a transition probability $\mathcal{T}$ giving the probability of stepping from state s at time t to state s' at time t+1 after having performed action a when in state s. To this transition is also associated a reinforcement signal (or reward) r_{t+1} describing how good was the result of action a when performed in state s. Formally, an MDP is thus completely defined by a 4-tuple $\{S, A, \mathcal{T}, \mathcal{R}\}$ where S is the state space, A is the action set, $\mathcal{T}$ is a transition probability distribution over the state space and $\mathcal{R}$ is the expected reward distribution. The couple $\{\mathcal{T}, \mathcal{R}\}$ defines the dynamics of the system:

$$\mathcal{T}_{ss'}^{a} = P\left(s_{t+1} = s' \mid s_t = s, a_t = a\right) \tag{1}$$

$$\mathcal{R}_{ss'}^{a} = E\left[r_{t+1} \mid s_t = s, a_t = a, s_{t+1} = s'\right] \tag{2}$$

These expressions assume that the Markov property is met, which means that the system's functioning is fully defined by its one-step dynamics and that its behavior from state *s* will be identical whatever the path followed before reaching *s*. To control a system described as an MDP (choosing actions to perform in each state), one would need a *strategy* or *policy* π mapping states to actions: π(*s*) = P(*a* | *s*) (or π(*s*) = *a* if the strategy is deterministic).

In this framework, a RL *agent* is a system aiming at optimally mapping states to actions, that is finding the best strategy π* so as to maximize, for each state, an overall return *R* which is a function (most often a discounted return is used i.e. a weighted sum of immediate rewards) of all the immediate rewards r_t.

$$\mathrm{R}^{\pi}(\mathrm{s}) = \mathrm{E}\left[\sum_{t=0}^{\infty} \gamma^{t} \mathrm{r}_{t} \mid \mathrm{s}_{0} = \mathrm{s}, \mathrm{a}_{t} = \pi(\mathrm{s}_{t})\right] \tag{3}$$

$$\pi^{*}(\mathrm{s}) = \underset{\pi}{\mathit{arg\,max}}\left[\mathrm{E}\left[\sum_{t=0}^{\infty} \gamma^{t} \mathrm{r}_{t} \mid \mathrm{s}_{0} = \mathrm{s}, \mathrm{a}_{t} = \pi(\mathrm{s}_{t})\right]\right] \tag{4}$$

where γ is a discount factor (0 < γ ≤ 1). If the probabilities of equations (1) and (2) are known, an analytical solution can be computed by resolving the Bellman equations using dynamic programming (Bertsekas 1995), otherwise the system has to learn the optimal strategy by a trial-and-error process.

To do so, a standard approach is to model the knowledge of the agent as a so-called Q-function mapping state-action pairs to an estimate of the expected cumulative reward. The optimal Q-function maps each state-action pair to its maximum expected cumulative rewards and the role of the agent can therefore be summarized as learning this function through interactions.

$$Q^{\pi}(s,a) = \int_{S} p(z|s,a)\Big(r(s,a,z) + \gamma Q^{\pi}(z,\pi(z))\Big)dz \tag{5}$$

Different techniques are described in the literature and in the following the Watkin's Q(λ) algorithm (Watkin 1989) will be used. This algorithm performs the following update after each interaction:

$$\hat{Q}(s,a) \leftarrow \hat{Q}(s,a) + \alpha\left(r + \gamma \max_{b \in A} \hat{Q}(s',b) - \hat{Q}(s,a)\right) \tag{6}$$

where α is a learning rate (0 < α ≤ 1). This algorithm has been proven to converge towards the optimal solution.

3. Human-machine dialogue and Markov decision process

A first requirement to use machine learning methods such as reinforcement learning for SDS optimization is to describe a man-machine dialogue in terms of random variables and probabilities. To do so, given the description of section 2.2, we adopt the dialogue manager point of view from which the interaction can probabilistically be described by the joint probability of the signals a_t, o_t and s_{t+1} given the history of the interaction (Pietquin 2005):

$$\begin{aligned} P(s_{t+1},o_t,a_t \mid s_t,n_t,a_{t-1},s_{t-1},n_{t-1},\ldots,a_0,s_0,n_0) &= \underbrace{P(s_{t+1} \mid o_t,a_t,s_t,n_t,a_{t-1},s_{t-1},n_{t-1},\ldots,a_0,s_0,n_0)}_{\text{Task Model}} \cdot \\ &\underbrace{P(o_t \mid a_t,s_t,n_t,a_{t-1},s_{t-1},n_{t-1},\ldots,a_0,s_0,n_0)}_{\text{Environment}} \cdot \underbrace{P(a_t \mid s_t,n_t,a_{t-1},s_{t-1},n_{t-1},\ldots,a_0,s_0,n_0)}_{\text{DM}} \end{aligned} \quad (7)$$

In (7), the *task model* term aims at describing the way the dialogue manager builds its internal state thanks to the perceived observation, the second term stands for the environment's response to the dialogue manager's stimulation, and the last stands for the dialogue manager decision process or strategy.

3.1 Markov property and random noise

As said in section 2.3, the Markov property has to be met so as to apply standard reinforcement learning methods for strategy optimization. In the case of a SDS, the Markov property implies that the dialogue manager choice about the communicative act a_t to choose at time t and the according transition probability for stepping to internal state s_{t+1} at time t+1 are only a function of the state s_t at time t and not of the history of interactions. It can easily be met by a judicious choice of the DM state representation, which should embed enough *information* about the history of the interaction into the current state description. Such a state representation is said *informational*.

This can be easily illustrated on a simple train ticket booking system. Using such a system, a customer can book a ticket by providing orally information about the cities of departure and arrival and a desired time of departure. Three bits of information (sometimes called *attributes*) have therefore to be transferred from the human user (or caller) to the system. The problem can be seen as filling a 3-slot form. From this, a very simple way to build the state space is to represent the dialogue state as a vector of three Boolean values (e.g. [dep arr time]) set to *true* if the corresponding attribute is considered as transferred to the system and to *false* otherwise. Table 1 shows an ideal dialogue for such an application with the associated dialogue state evolution.

Speaker	Spoken Utterance	Dialogue state
System	Hello, how may I help you?	[*false false false*]
User	I'd like to go to Edinburgh.	
System	What's your departure city?	[*false true false*]
User	I want to leave from Glasgow.	
System	When do you want to go from Glasgow to Edinburgh?	[*true true false*]
User	On Saturday morning.	
System	Ok, seats are available in train n° xxx …	[*true true true*]

Table 1. Ideal dialogue in a train ticket booking application

To assume the Markov property is met using this state representation, one have make the assumption that the system adopts the same behaviour whatever the order in which the slots where filled (and by the way, whatever the values of the attributes). The Markov assumption is also made about the environment; that is the user behaves the same whatever the filling order as well. These are of course strong assumptions but we will see later that they lead to satisfactory results.

Finally, most often the noise is considered as being random so as to have independence between n_t and n_{t-1}. Eq. (5) then simplifies as follow:

$$P(s_{t+1},o_t,a_t \mid s_t,n_t) = \underbrace{P(s_{t+1} \mid o_t,a_t,s_t,n_t)}_{\text{Task Model}} \cdot \underbrace{P(o_t \mid a_t,s_t,n_t)}_{\text{Environment.}} \cdot \underbrace{P(a_t \mid s_t,n_t)}_{\text{DM}} \quad (8)$$

3.2 Dialogue management as an MDP

From paragraph 2.2, the observation o_t can be regarded as the result of the processing of the DM dialog act a_t by its *environment*. This point of view helps putting dialogue management optimization into the MDP framework. As depicted on Fig. 2, a task-oriented (or goal-directed) man-machine dialogue can be regarded as a turn-taking process in which a user and a dialogue manager exchange information through different channels processing speech inputs and outputs (ASR, TTS ...). The dialogue manager's action (or dialogue act) selection *strategy* has to be optimized; the dialogue manager should thus be the learning *agent.*

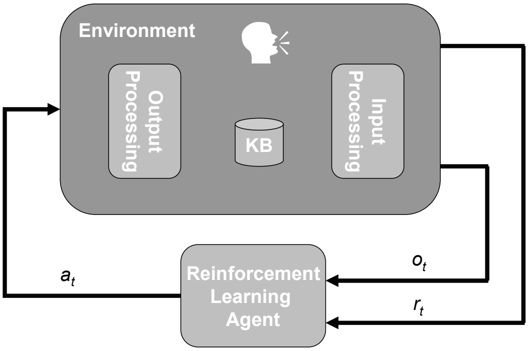


Fig. 4. Dialogue management as an MDP

The *environment* modeled by the RL agent as an MDP includes everything but the dialogue manager (see Fig. 4), i.e. the human user, the communication channels (ASR, TTS …), and any external information source (database, sensors etc.). In this context, at each turn *t* the dialogue manager has to choose an *action* a_t according to its interaction *strategy* so as to complete the task it has been designed for. The RL agent has therefore to choose an action among greetings, spoken utterances (constraining questions, confirmations, relaxation, data presentation etc.), database queries, dialogue closure etc. They result in a response from the DM environment (user speech input, database records etc.), considered as an observation o_t, which usually leads to a DM *internal state* update according to the task model (Eq. 8).

3.3 Reward function

To entirely fit to the Reinforcement Learning formalism, the previous description is still missing a *reward signal* r_t. Different ideas could lead to the building of this signal such as the amount of money saved by using a SDS instead of having human operators or the number of people hanging off before the end of the interaction etc. Singh *et al* in 1999 proposed to use the contribution of an action to the user's satisfaction. Although this seems very subjective, some studies have shown that such a reward could be approximated by a linear combination of the task completion (TC) and objective measures c_i related to the system performances. It is the PARADISE paradigm proposed in Walker *et al* 1997:

$$r_t = \alpha \cdot \mathcal{N}(TC) - \sum_i w_i \cdot \mathcal{N}(c_i), \tag{9}$$

where N is a Z-score normalization function that normalises the results to have mean 0 and standard deviation 1 and w_i are non-zero weights. Each weight (α and w_i) thus expresses the relative importance of each term of the sum in the performance of the system. There are various ways to associate an objective measure to the task completion. For example the kappa (κ) coefficient (Carletta 1996) is defined as:

$$\kappa = \frac{P(A) - P(E)}{1 - P(E)}, \tag{10}$$

where $P(A)$ is the proportion of correct interpretations of user's utterances by the system and $P(E)$ is the proportion of correct interpretations occurring by chance. One can see that κ = 1 when the system performs perfect interpretation ($P(A) = 1$) and κ = 0 when the all the correct interpretations were obtained by chance ($P(A) = P(E)$).

The weights α and w_i are obtain by asking a large number of users to use a prototype system and to answer a satisfaction survey containing around 9 statements on a five-point Likert scale. The overall satisfaction is computed as the mean value of collected ratings. The objective costs c_i are measured during the interaction. A Multivariate Linear Regression is then applied using the results of the survey as the dependent variable and the weights as independent variables. In practice, the significant performance measures c_i are mainly the duration of the dialogue and the ASR and NLU performances.

3.4 Partial observability

When a direct mapping between states and observations exists, building the task model (eq. 8) is straightforward. Yet, it is rarely the case that the observations can directly be translated into dialogue states. Indeed, the real dialogue state (which we have chosen informational) at time t is related to the information the user *intended* to transmit to the system until time t during the interaction. The statistical speech recognition and understanding systems processing the user speech inputs are error prone and it can occur that the observation doesn't contain only the information meant by the user but a probability distribution over a set of possible bits of information. Indeed, as said before, the output of a speech recognition system can be a list of N word sequences (named N-best list), each of them being associated with a confidence level. This can be considered as a probability of the word sequence being correct given the spoken utterance (and maybe the context). This N-bests list serves as an input to the natural language understanding module which in turn provides a list of concept sequences associated to confidence levels.

This is typically what happens in partially observable environments where a probability distribution is drawn over possible states given the observations. An observation model is therefore usually required. It is what we have called the task model in eq. 8 which can be a real probability distribution. For this reason, emerging research is focused on the optimization of spoken dialogue systems in the framework of Partially Observable Markov Decision Processes (POMDPs) (Poupart *et al* 2005, Young 2006)

4. Learning dialogue policies using simulation

Using the framework described previously, it is theoretically possible to automatically learn spoken dialogue policies allowing natural conversation between human users and computers. This learning process should be realised online, through real interactions with users. One could even imagine building the reinforcement signal from direct queries to the user about his/her satisfaction after each interaction (Fig. 5).

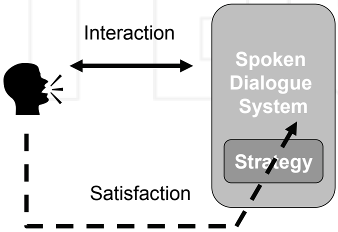


Fig. 5. Ideal learning process

For several reasons, direct learning through interactions is made difficult. First, a human user would probably react badly to some of the exploratory actions the system would choose since they might be completely incoherent. Anyway a very large number of interactions are required (typically tens of thousands of dialogues for standard dialogue systems) to train such a system. This is why data driven learning as been proposed so as to take advantage of existing databases for bootstrapping the learning process. Two methods were initially investigated: learning the state transition probabilities and the reward distribution from data (Singh *et al*, 1999) or learning parameters of a simulation environment mainly reproducing the behaviour of the user (Levin *et al* 2000). The second method is today preferred (Fig. 6). Indeed, whatever the data set available, it is unlikely that it contains every possible state transitions and it allows exploring the entire spaces. Dialogue simulation is therefore necessary for expanding the existing data sets and learning optimal policies. Another track of research is dealing with generalization to unseen situation. In this case, instead of simulating unseen situations, machine learning generalization methods are used to compute a Q-function over the entire state space with only a finite set of samples (Henderson *et al* 2005).

Most often, the dialogue is simulated at the intention level rather than at the word sequence or speech signal level, as it would be in the real world. An exception can be found in (Lopez Cozar *et al* 2003). Here, we regard an intention as the minimal unit of information that a dialogue participant can express independently. Intentions are closely related to concepts, speech acts or dialogue acts. For example, the sentence "I'd like go to Edinburgh" is based on the concept go(Edinburgh). It is considered as unnecessary to model environment behavior at a lower level, because strategy optimization is a high level concept. Additionally, concept-based communication allows error modeling of all the parts of the system, including natural language understanding (Pietquin & Renals 2002, Pietquin & Dutoit 2006b). More pragmatically, it is simpler to automatically generate concepts compared with word sequences (and certainly speech signals), as a large number of utterances can express the same intention while it should not influence the dialogue manager strategy. Table 2 describes such a simulation process. The intentions have been expanded in the last column for comprehensiveness purposes. The signals column refers to notations of section 2.2.

Signals	**Intentions**	**Expanded Intentions**
$\mathbf{sys}_0$	greeting	*Hello! How may I help you?*
$\mathbf{u}_0$	arr_city = 'Paris'	I'd like to go to Paris.
$\mathbf{sys}_1$	const(arr_time)	*When do you prefer to arrive?*
$\mathbf{u}_1$	arr_time = '1.00 PM'	I want to arrive around 1 PM.
$\mathbf{sys}_2$	rel(arr_time)	*Don't you prefer to arrive later?*
$\mathbf{u}_2$	rel = *false*	No.
$\mathbf{sys}_3$	conf(arr_city)	*Can you confirm you want to go to Paris?*
$\mathbf{u}_3$	conf = *true*	Yes !
…	…	…
…	…	…

Table 2. Simulated dialogue at the intention level ('const' stands for constraining question, 'rel' for relaxation and 'conf' for confirmation)

This approach requires modelling the environment of the dialogue manager as a stochastic system and to learn the parameters of this model from data. It has been a topic of research since the early 2000's (Levin *et al* 2000, Scheffler & Young 2001, Pietquin 2004). Most of the research is now focused on simulating the user (Georgila *et al* 2005, Pietquin 2006a, Schatzmann *et al* 2007a) and assessing the quality of a user model for training a reinforcement learning agent is an important track (Schatzmann *et al 2005,* Rieser & Lemon 2006, Georgila *et al* 2006). Modelling the errors introduced by the ASR and NLU systems is also a major topic of research (Scheffler & Young 2001, Lopez Cozar *et al* 2003, Pietquin & Beaufort 2005, Pietquin & Dutoit 2006b).

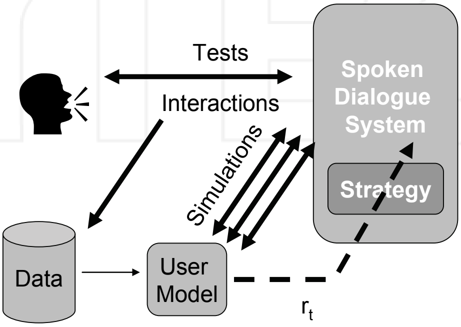


Fig. 6. Learning via simulation

4.1 Probabilistic user simulation

According to the conventions of Fig. 2 and omitting the *t* indices, the user behavior is ruled by the following joined probability that can be factored and simplified:

$$
\begin{aligned}
P(u,g,k|sys,a,s,n) &= \underbrace{P(k\mid sys,a,s,n)}_{\text{Knowledge Update}}\cdot \underbrace{P(g\mid k,sys,a,s,n)}_{\text{Goal Modification}}\cdot \underbrace{P(u\mid g,k,sys,a,s,n)}_{\text{User Output}}\\
&= \underbrace{P(k\mid sys,s,n)}_{\text{Knowledge Update}}\cdot \underbrace{P(g\mid k)}_{\text{Goal Modification}}\cdot \underbrace{P(u\mid g,k,sys,n)}_{\text{User Output}}
\end{aligned}
$$

These terms emphasize on the relation existing between the user's utterance production process and his/her goal and knowledge, themselves linked together. The knowledge can be modified during the interaction through the speech outputs produced by the system. Yet, this modification of the knowledge is incremental (it is an update) and takes into account the last system utterance (which might be misunderstood, and especially in presence of noise) and the previous user's knowledge state. This can be written as follow with $\boldsymbol{k}^{-}$ standing for k_{t-1}:

$$
\begin{aligned}
P(k\mid sys,s,n) &= \sum\nolimits_{k^-} P(k\mid k^-,sys,s,n)\cdot P(k^-\mid sys,s,n)\\
&= \sum\nolimits_{k^-} P(k\mid k^-,sys,n)\cdot P(k^-\mid s)
\end{aligned}
$$

The parameter of this model can be learnt from data. In (Pietquin & Dutoit, 2006b), this model serves as a basis to define a Dynamic Bayesian Network (DBN) (Fig. 7). This allows using standard DBN tools to simulate a user model and to learn the parameters from data.

Although the user's knowledge k^- is not directly dependent of the system state *s*, we kept this dependency in our description so as to be able to introduce a mechanism for user knowledge inference from system state because it is supposed to contain information about the history of the dialogue. This mechanism can actually be used to introduce grounding (Clarck et Shaefler, 1989) subdialogs in the interaction so as to obtain a good connection between the user's understanding of the interaction and the system view of the same interaction (Pietquin, 2007).

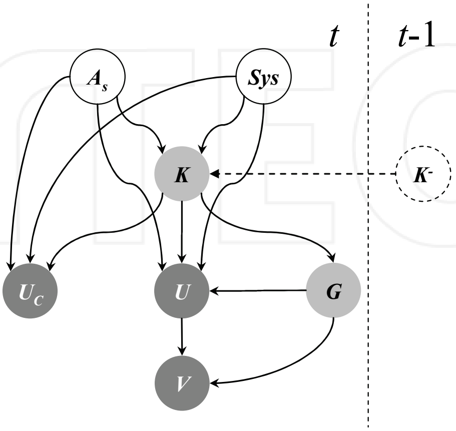


Fig. 7. DBN-based user model

4.2 Attribute-Value variable representation

It is quite unclear how to model each variable present in this description (such as u_t, sys_t, g_t etc.) for computer-based HMD simulation. As said before, it is often argued that *intention-based* communication is sufficient to internally model dialogs. Variables can then be regarded as finite sets of abstract concepts, related to the specific task, that have to be manipulated along the interactions by the SDS and the user. For this reason, we opted for a variable representation based on Attribute-Value (AV) pairs. This representation allows very high-level considerations (attributes are regarded as concepts) while values (particular values for the concepts) allow to some extent to come back to lower levels of communication. This variable description is founded on an Attribute-Value-Matrix (AVM) representation of the task (Walker *et al*, 1999)

Each communicative act is then symbolized by a set of AV pairs. From now on, we will denote ***A*** the set of possible attributes (concepts) according to the task, and by ***V*** the set of all possible values. The system utterances *sys* are then modeled as sets of AV pairs in which the attribute set will be denoted $\boldsymbol{Sys}=\{sys^{\sigma}\} \subset A$ and the set of possible values for each attribute sys^{σ} will be denoted $\boldsymbol{V^{\sigma}} = \{ \mathrm{v}_{\mathrm{i}}^{\sigma} \} \subset V$. The system utterance attribute set contains a special attribute $\boldsymbol{A_S}$ which values define the type of the embedded act. Allowed types can be constraining questions, relaxing prompts, greeting prompts, assertions, confirmation queries, etc. The user's utterance u is modeled as a set of AV pairs (transmitted to the ASR model) in which attributes belong to $\boldsymbol{U} = \{u^{\upsilon}\} \subset A$ and the set of possible values for u^{υ} is $\boldsymbol{V^{\upsilon}} = \{ \mathrm{v}_{\mathrm{i}}^{\upsilon} \} \subset V$. The user's utterance attribute set contains a special attribute $\boldsymbol{C_U}$ which value is a Boolean indicating whether the user wants to close the dialog or not. The ASR process results in an error-prone set of AV pairs w which is in turn processed and possibly modified by the NLU model. This process provides a new AV pair set c, which is part of the observation o. The user's goal $\boldsymbol{G} = \{[g^{\gamma}, \mathrm{gv}_{\mathrm{i}}^{\gamma}]\}$ and the user's knowledge $\boldsymbol{K} = \{[k^{\kappa}, \mathrm{kv}_{\mathrm{i}}^{\kappa}]\}$ are also AV pair sets where g^{γ} and k^{κ} are attributes and where $\mathrm{gv}_{\mathrm{i}}^{\gamma}$ and $\mathrm{kv}_{\mathrm{i}}^{\kappa}$ are values.

5. Experiment

This model was developed in the aim of being used in an optimal dialog strategy learning process. We therefore show here a use case of dialog simulation for Reinforcement-Learning (RL) agent training on a simple form-filling dialog task. To do so, a reward function (or reinforcement signal) $\boldsymbol{r_t}$ has to be defined. This reward provides information about the quality of each DM decision of performing an action a when in state s at time t. It is generally considered that the contribution of each action to the user's satisfaction is the most suitable reward function (Singh *et al*, 1999). According to (Walker *et al*, 1997), the major contributors to user's satisfaction are the dialog time duration (which can be approximated by the number of dialog turns ***N***), the ASR performances (which we will approximate by a confidence level ***CL*** as in (Pietquin & Renals, 2002) and the task completion (***TC***). For this reason, we chose a reward function of the form:

$$r_t = w_{TC} \cdot TC + w_{CL} \cdot CL - w_N \cdot N$$

where w_x are positive tunable weights.

The task is a simplified version of a train ticket booking system that aims at delivering train tickets corresponding to a particular travel. Users are invited to provide information about the departure city (over 50 possible options) and time (over 24 possible options) as well as the destination city and time. The desired class (2 options) is also requested. Table 3 shows the task structure, the user's goal structure (AV pairs) and the knowledge structure which will be simply a set of counters associated to each goal AV pair and incremented each time the user answers to a question related to a given attribute during the dialog. The task completion is therefore measured as a ratio between the common values in the goal and the values retrieved by the system after the dialog session. The simulation environment includes the DBN user model, and an ASR model like in (Pietquin & Renals, 2002).

The RL paradigm requires the definition of a state space. It will be defined by a set of state variables which are 5 Booleans (one for each attribute in the task) set to *true* when the corresponding value is known, 5 status Booleans set to *true* if the corresponding value is confirmed and 5 binary values indicating whether the Confidence Level (CL) associated to the corresponding value is *high* or *low*. Every combination is not possible and the state space size is therefore of 52 states. The DM will be allowed 5 action types: greeting, open question (about more than 1 attribute), closed question (about only 1 attribute), explicit confirmation, closing.

Task		User Goal (*G*)		Knowledge (*K*)	
Attributes (A)	#V	Att.	Value	Count	init
dep	50	g^{dep}	Glasgow	k^{dep}	0
dest	50	g^{dest}	Edinburgh	k^{dest}	0
t_dep	24	g^{t_dep}	8	k^{t_dep}	0
t_dest	24	g^{t_dest}	12	k^{t_dest}	0
class	2	g^{class}	1	k^{class}	0

Table 3. AV representation of the task

Performance	
N_U	TC
5.39	0.81

Strategy				
greet	constQ	openQ	expC	close
1.0	0.85	1.23	1.31	1.0

Table 4. Experimental results

The results of the learning process on 10^5 dialogs shown in Table 4 can be interpreted as follow. This experiment shows that, in our model, the user's satisfaction relies as much on the duration time as on the task completion. Thus dialogues are short, but task completion is not optimal since one attribute is often missing in the presented data (one of the cities in general). There are more open-ended questions than constraining questions. Actually, constraining questions are present because sometimes only one argument is missing and there is no need of an open-ended question to retrieve it. Yet, there are explicit confirmations because the task completion is a factor of user satisfaction. It actually illustrates well the trade-off between task completion and time duration. This behaviour can be tuned by changing the parameters of our user model for example.

6. Conclusion

In this chapter, a formal probabilistic description of human-machine dialogues was described. This description allowed putting the optimization of spoken dialogue strategies in the framework of reinforcement learning. Reinforcement learning designates a very data-demanding class of machine learning methods. This is a major problem for SDS optimization since collecting and annotating data is very difficult. To solve this problem of data sparsity, dialogue simulation techniques are commonly used. A specific simulation framework based on a probabilistic description of the user's behavior has been described. It can easily be translated into a dynamic Bayesian network and use the standard parameter learning and inference tools. The reinforcement learning framework also requires the definition of a reward function associating a numerical number to each system action. To do so, the PARADISE framework using multivariate regression has been described. To summarize, this chapter has shown that a large number of machine learning methods can be used in the context of spoken dialogue optimization. Among these techniques, reinforcement learning, Bayesian inference and multivariate regression are very common.

7. Future works

Statistical machine learning for spoken dialogue strategies optimization is an emerging area of research and lots of issues still remain. One of the first, which is common to a lot of reinforcement learning applications, is to find tractable algorithms for real size dialogue systems. The standard RL algorithms are indeed suitable for small tasks such as described in section 5. Yet real applications can exhibit up to several million of states, possibly with continuous observations (Williams *et al* 2005). Supervised learning (Henderson *et al* 2005) and hierarchical learning (Cuayáhuitl *et al* 2007) have been recently proposed to tackle this problem.

In this chapter, we have essentially considered the problem of completely observable systems. But as said in paragraph 3.4, a spoken dialogue system should be considered as partially observable, because of error prone speech processing sub-systems. Research on POMDP for SDS optimization are reported in (Poupart *et al* 2005, Young 2006), yet a lot of work is still necessary to anchor SDS in real life.

Spoken dialogue simulation is also the topic of ongoing research. Different approaches are being studied such as the recently proposed agenda-based user model (Schatzmann *et al* 2007b) that can be trained by an Expectation-Maximisation algorithm from data, or user models based on dynamic Bayesian networks (Pietquin & Dutoit 2006a) such as those presented in this chapter. One of the major argument against the current simulation methods is the lack of assessment methods even though some work can be cited (Schatzmann *et al* 2005, Georgila *et al* 2006, Rieser & Lemon 2006).

On another hand, it might be interesting to see how to use learned strategies to help human developers to design optimal strategies. Indeed, the solution may be in computer-aided design more than fully automated design (Pietquin & Dutoit 2003).

The ultimate aim of this research area is to design a complete data-driven dialogue system using an end-to-end probabilistic framework, from speech recognition to speech synthesis systems automatically trained on real data, is probably the next step (Lemon & Pietquin 2007).

8. Acknowledgement

The research presented in this chapter has been funded by the 'First Europe' program of the Belgian Walloon Region, the SIMILAR European Network of Excellence and the French Lorraine Region.

9. References

Allen, J. (1994) *Natural Language Understanding*, Benjamin Cummings, 1987, Second Edition, 1994.

Bertsekas, D. P. (1995). *Dynamic Programming and Optimal Control*. Athena Scientific, 3rd edition.

Carletta J. (1996), Assessing Agreement on Classification Tasks: the Kappa Statistic. *Computational Linguistics*, 22(2), 1996, 249-254.

Clarck H. and Schaefer E., "Contributing to discourse," Cognitive Science, vol. 13, pp. 259–294, 1989.

Cuayáhuitl, H.; Renals, S.; Lemon, O. and Shimodaira, H. (2007) Hierarchical Dialogue Optimization Using Semi-Markov Decision Processes, in *Proceedings of International Conference on Speech Communication (Interspeech'07)*, Anvers (Belgium), 2007.

Dutoit, T., *An Introduction to Text-To-Speech Synthesis*. Kluwer Academic Publishers, Dordrecht, ISBN 0-7923-4498-7, 1997.

Frampton, M. & Lemon O. (2006). Learning more effective dialogue strategies using limited dialogue move features, in *Proceedings of ACM*, 2006.

Georgila, K.; Henderson, J. and Lemon, O. (2005). Learning User Simulations for Information State Update Dialogue Systems, in *Proceedings of International Conference on Speech Communication (Interspeech'05)*, Lisbon (Portugal) 2005.

Georgila, K.; Henderson, J. and Lemon, O. (2006) User simulation for spoken dialogue systems: Learning and evaluation, in *Proceedings of International Conference on Speech Communication (Interspeech'06)*, Pittsburgh, 2006.

Graesser, A.; VanLehn, K.; Rosé, C.; Jordan, P. & Harter, D. (2001) Intelligent Tutoring Systems with Conversational Dialogue. in *AI Magazine* vol. 22(4) , 2001, pp. 39-52.

Henderson, J.; Lemon, O. and Georgila, K. (2005) Hybrid Reinforcement/Supervised Learning for Dialogue Policies from COMMUNICATOR data, in *Proceedings of the IJCAI workshop on Knowledge and Reasoning in Practical Dialogue Systems*, 2005, pp. 68–75.

Lemon, O. & Pietquin, O. (2007). Machine learning for spoken dialogue systems, in *Proceedings of the European Conference on Speech Communication and Technologies (Interspeech'07)*, Anvers (Belgium), August 2007.

Levin, E.; Pieraccini, R. & Eckert, W. (1997). Learning dialogue strategies within the Markov decision process framework, in *Proceedings of the International Workshop on Automatic Speech Recognition and Understanding* (ASRU'97), December 1997.

Levin, E.; Pieraccini, R. and Eckert, W. (2000). A stochastic model of human-machine interaction for learning dialog strategies, in *IEEE Transactions on Speech and Audio Processing*, vol. 8, no. 1, pp. 11–23, 2000.

Lopez-Cozar, R.; de la Torre, A.; Segura, J. and Rubio, A. (2003) Assesment of dialogue systems by means of a new simulation technique, in *Speech Communication*, vol. 40, no. 3, pp. 387–407, May 2003.

Pietquin, O. and Renals, S. (2002). Asr system modelling for automatic evaluation and optimization of dialogue systems, in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP'02)*, Orlando, (USA, FL), May 2002.

Pietquin, O. and Dutoit, T. (2003). Aided Design of Finite-State Dialogue Management Systems, in *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME 2003)*, Baltimore (USA, MA), 2003.

Pietquin, O. (2004). *A Framework for Unsupervised Learning of Dialogue Strategies*, Presses Universitaires de Louvain, ISBN : 2-930344-63-6, 2004.

Pietquin, O. (2005). A probabilistic description of man-machine spoken communication, in *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME'05)*, Amsterdam (The Netherlands), July 2005.

Pietquin, O., Beaufort, R. (2005). Comparing ASR Modeling Methods for Spoken Dialogue Simulation and Optimal Strategy Learning. In *Proceedings of Interspeech/Eurospeech 2005*, Lisbon, Portugal (2005)

Pietquin, O. (2006a) Consistent goal-directed user model for realistic man-machine task-oriented spoken dialogue simulation, in *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME'06)*, Toronto, Canada, July 2006.

Pietquin, O. (2006b). Machine learning for spoken dialogue management : an experiment with speech-based database querying, in *Artificial Intelligence : Methodology, Systems and Applications*, J. Euzenat and J. Domingue, Eds., vol. 4183 of Lecture Notes in Artificial Intelligence, pp. 172–180. Springer Verlag, 2006.

Pietquin, O. & Dutoit, T. (2006a). A probabilistic framework for dialog simulation and optimal strategy learning, in *IEEE Transactions on Audio, Speech and Language Processing*, vol. 14, no. 2, pp. 589–599, March 2006.

Pietquin, O. and Dutoit, T. (2006b). Dynamic Bayesian networks for NLU simulation with applications to dialog optimal strategy learning, in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP'06)*, May 2006.

Pietquin, O. (2007), Learning to Ground in Spoken Dialogue Systems. *In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2007)*, IV:165-168, Hawaii (USA), 2007

Poupart, P.; Williams, J. & Young, S. (2006). Partially observable Markov decision processes with continuous observations for dialogue management, in *Proceedings of the SigDial Workshop (SigDial'06)*, 2006.

Rabiner, L. & Juang, B.H. (1993). *Fundamentals of Speech Recognition*, Prentice Hall, Signal Processing Series, 1993.

Reiter, E. & Dale, R. (2000) *Building Natural Language Generation Systems*, Cambridge University Press, Cambridge, 2000.

Rieser, V. and Lemon, O. (2006) Cluster-based user simulations for learning dialogue strategies and the super evaluation metric, in *Proceedings of Interspeech/ICSLP*, 2006.

Schatzmann, J.; Georgila, K. and Young, S. (2005) Quantitative evaluation of user simulation techniques for spoken dialogue systems, in *Proceedings of the SIGdial'05 Workshop*, September 2005.

Schatzmann, J.; Weilhammer, K.; Stuttle, M. and Young, S. (2007a) A survey of statistical user simulation techniques for reinforcement-learning of dialogue management strategies, in *Knowledge Engineering Review* 21(2): 97-126, 2007.

Schatzmann, J.; Thomson, B. and Young., S. (2007b). Statistical User Simulation with a Hidden Agenda. In *Proceedings of the 8th SigDIAL Workshop*, Antwerp, 2007.

Scheffler, K. & Young, S. (2001). Corpus-based dialogue simulation for automatic strategy learning and evaluation, in *Proc. NAACL Workshop on Adaptation in Dialogue Systems*, 2001.

Singh, S.; Kearns, M.; Litman, D. & Walker, M. (1999), Reinforcement learning for spoken dialogue systems, in *Proceedings of NIPS'99*, 1999.

Young, S. (2006). Using POMDPs for dialog management, in *Proceedings of the 1st IEEE/ACL Workshop on Spoken Language Technologies (SLT'06)*, 2006.

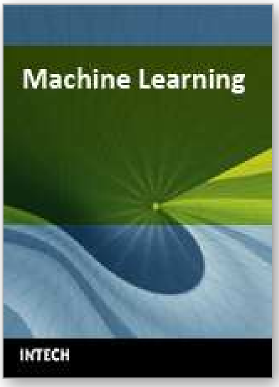
Young, S.; Schatzmann, J.; Weilhammer, K. & Ye, H. (2007). The hidden information state approach to dialog management, in *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP'07)*, April 2007.

Walker, M.; Litman, D.; Kamm, C. & Abella, A. (1997). PARADISE: A Framework for Evaluating Spoken Dialogue Agents. in *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics*, Madrid, Spain (1997) 271-280.

Watkins, C. (1989). *Learning from delayed rewards*. PhD thesis, Psychology Department, Cambridge University, Cambridge, England, 1989.

Williams, J. & Young, S. (2005). Scaling up POMDPs for dialogue management: the summary POMDP method, in *Proceedings of the IEEE workshop on Automatic Speech Recognition and Understanding (ASRU'05)*, 2005.

Williams, J.; Poupart, P. and Young, S. (2005). Partially Observable Markov Decision Processes with Continuous Observations for Dialogue Management, in *Proceedings of the 6th SigDial Workshop*, Lisbon (Portugal), 2005.



Machine Learning

Edited by Abdelhamid Mellouk and Abdennacer Chebira

ISBN 978-953-7619-56-1
Hard cover, 450 pages
Publisher InTech
Published online 01, January, 2009
Published in print edition January, 2009

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821