Task Completion Transfer Learning for Reward Inference

Article · July 2014

CITATIONS
3

READS
45

3 authors:

Layla El Asri
Maluuba
18 PUBLICATIONS **68** CITATIONS
SEE PROFILE

Romain Laroche
Microsoft Maluuba
58 PUBLICATIONS **185** CITATIONS
SEE PROFILE

Olivier Pietquin
Google DeepMind
203 PUBLICATIONS **1,330** CITATIONS
SEE PROFILE

Task Completion Transfer Learning for Reward Inference

Layla El Asri[1,2], **Romain Laroche**[1], **Olivier Pietquin**[3]
[1]Orange Labs, Issy-les-Moulineaux, France
[2] UMI 2958 (CNRS - GeorgiaTech), France
[3]University Lille 1, LIFL (UMR 8022 CNRS/Lille 1) - SequeL team, France
layla.elasri@orange.com, romain.laroche@orange.com, olivier.pietquin@univ-lille1.fr

Abstract

Reinforcement learning-based spoken dialogue systems aim to compute an optimal strategy for dialogue management from interactions with users. They compare their different management strategies on the basis of a numerical reward function. Reward inference consists of learning a reward function from dialogues scored by users. A major issue for reward inference algorithms is that important parameters influence user evaluations and cannot be computed online. This is the case of task completion. This paper introduces Task Completion Transfer Learning (TCTL): a method to exploit the exact knowledge of task completion on a corpus of dialogues scored by users in order to optimise online learning. Compared to previously proposed reward inference techniques, TCTL returns a reward function enhanced with the possibility to manage the online non-observability of task completion. A reward function is learnt with TCTL on dialogues with a restaurant seeking system. It is shown that the reward function returned by TCTL is a better estimator of dialogue performance than the one returned by reward inference.

Introduction

In a Spoken Dialogue System (SDS), the dialogue manager controls the behaviour of the system by choosing which dialogue act to perform according to the current context. Adaptive SDS now integrate data-driven statistical methods to optimise dialogue management. Among these techniques, Reinforcement Learning (RL) (Singh et al. 1999) compares and assesses management strategies with a numerical reward function. Since this function serves as a dialogue quality evaluator, it must take into account all the different variables which come into play in dialogue success. SDS evaluation might be used to discover these variables (Lemon and Pietquin 2012).

Evaluation campaigns on many disparate systems have enabled to highlight common Key Performance Indicators (KPI) such as task completion, dialogue duration and speech recognition rejection/error rate (Walker et al. 1997b; Larsen 2003; Lemon and Pietquin 2012). Therefore, a reward function would ideally integrate and be able to estimate online all these KPI. Nevertheless, the correctness of speech recognition and the task completion cannot always be accurately estimated online. We focus in this paper on circumventing the task completion problematic.

Walker *et al.* (Walker et al. 1997b) designed a PARAdigm for DIalogue System Evaluation (PARADISE), which models SDS performance as the maximisation of task completion and the minimisation of dialogue costs such as dialogue duration or the number of rejections from speech recognition. Multiple linear regression has been proposed to compute an estimator of SDS performance. Dialogue costs are automatically computed from dialogue logs and task completion is measured with the κ statistic (Cohen 1960). This estimator has been used as a reward function (Walker, Fromer, and Narayanan 1998; Rieser and Lemon 2011).

Information-providing systems are often built as slot-filling systems (Raux et al. 2003; Lemon et al. 2006a; Chandramohan et al. 2011). In the case of these systems, task completion is measured by the number of correctly filled slots. The κ statistic counts this number and adjusts it with the probability that the correct information was obtained by chance. This statistic cannot be computed online because, for each dialogue, it compares the values of the attributes (*e.g* location, price, type of food for a restaurant-seeking SDS) intended by the user to the ones understood by the SDS. When user intention is unknown, one cannot check the validity of the information provided by the SDS. In this context, one way to estimate the level of task achievement is to count the number of slots that were confirmed by the user during the dialogue. Nevertheless, this does not provide an exact measure of task completion so some dialogues might still be ill-evaluated.

Another common type of dialogue system is the utilitarian one. These systems are built to achieve a precise task like scheduling an appointment (Laroche et al. 2011; El Asri et al. 2014) or controlling some devices (Möller et al. 2004). It is also difficult in this context to estimate task completion with accuracy. For instance, concerning the appointment scheduling task, it was observed on scenario-based dialogues that some users had booked an appointment during a time slot when they were supposed to be busy. Because the scenario was known, the task was not considered to have been completed by the system but without knowing the user calendar, this outcome would have been impossible to discern.

All in all, in most cases, it is difficult to measure the task

completion of an online operating SDS. This paper proposes to use RL to improve task completion estimation accuracy. The technique introduced in this paper is in the same line as the RL research topic known as transfer learning. Transfer learning aims to use former training on a specific task to perform a related but different task (Taylor and Stone 2009). We introduce Task Completion Transfer Learning (TCTL), a technique that transfers training on a corpus of evaluated dialogues where task completion is known, to online learning, where it is not.

Reward inference computes a reward function from a corpus of evaluated dialogues. TCTL is based on a previously presented reward inference algorithm named reward shaping (El Asri, Laroche, and Pietquin 2012; 2013). TCTL temporarily includes task completion in the dialogue state space and learns a policy $\dot{\pi}$ which optimises user evaluation on this space. $\dot{\pi}$ is then used to adjust the reward function inferred by reward shaping.

TCTL is applied to a simulated corpus of dialogues with a restaurant-seeking dialogue system. The reward function learnt with reward shaping is compared to the one learnt with TCTL. These two functions provide an estimation of dialogue performance. It is shown that the estimation given by TCTL is closer to the real performance than the one given by reward shaping by comparing the rank correlation coefficients on the simulated dialogues.

Background

The stochastic decision process of dialogue management is implemented as a Markov Decision Process (MDP). An MDP is a tuple (S, A, T, R, γ) where S is the state space, T are the transition probabilities modelling the environmental dynamics: $\forall\ (s_t, a_t, s_{t+1}),\ T(s_t, a_t, s_{t+1}) = P(s_{t+1} \mid s_t, a_t)$, R is the reward function and $\gamma \in]0, 1[$ is a discount factor. A similar MDP without a reward function will be noted MDP$\backslash R$.

Reward shaping gives an immediate reward $R_t = R(s_t, s_{t+1})$ to the system for each transition (s_t, s_{t+1}). Time is measured in number of dialogue turns, each dialogue turn being the time elapsed between two consecutive results of Automatic Speech Recognition (ASR). The return r_t is the discounted sum of immediate rewards received after dialogue turn t: $r_t = \sum_{k \geq 0} \gamma^k R_{t+k}$. A deterministic policy π maps each state s to a unique action a. Under a policy π, the value of a state s is the expected return following π starting from state s: $V^\pi(s) = E[r_t \mid s_t = s, \pi]$. The Q-value of a state-action couple (s, a) under π is $Q^\pi(s, a) = E[r_t \mid s_t = s, a_t = a, \pi]$.

The aim of dialogue management is to compute an optimal deterministic policy π^*. π^* maximises the expected return for all dialogue states: $\forall\ \pi, \forall\ s, V^{\pi^*}(s) \geq V^\pi(s)$. The optimal policy for a given MDP might not be unique but the optimal policies share the same value functions.

Related work

The reward inference problem described in Definition 1 has been studied for SDS optimisation (Walker et al. 1997b; El Asri, Laroche, and Pietquin 2012; Sugiyama, Meguro, and Minami 2012).

Definition 1 (Reward inference) *Infer a reward function from a corpus of N dialogues $(D^i)_{i \in 1..N}$ among which p dialogues have been manually evaluated with a performance score $P^i \in \mathbb{R}$.*

These techniques have a different manner of dealing with task completion. Walker *et. al* (Walker et al. 1997b) propose an estimator of task completion computing the κ statistic. In (El Asri, Laroche, and Pietquin 2012), the relevant features for computing the rewards are directly included in the state space and task completion is handled in final states. In (Sugiyama, Meguro, and Minami 2012), a reward function is not designed for a goal-oriented but a conversational system. Reward inference is done by preference-based inverse reinforcement learning: the algorithm learns a reward function which follows the same numerical order as the performance scores P_i.

In (El Asri, Laroche, and Pietquin 2012), an algorithm named reward shaping was proposed to solve the reward inference problem. This method is recalled in Algorithm 1. Reward shaping returns the reward function R^{RS} in Equation 1.

Algorithm 1 Reward shaping algorithm

Require: A set of evaluated dialogues $D_1, ..., D_N$ with performance scores $P_1, ..., P_N$; a stopping criterion ϵ

for all $D_i \in D_1, .., D_N$ **do**
  for all decision $d_t \in D_i$ (score P_i) **do**
    Compute the return $r_t = \gamma^{-t} P_i$
  end for
end for
for all (s, a) **do**
  Update the state-action value function: $Q^{\pi_0}(s, a)$
end for
Update the policy: $\pi_1(s) = \text{argmax}_a\, Q^{\pi_0}(s, a)$
repeat
  for all s **do**
    Update the estimated performance $\hat{P}^{\pi_k}(s) = E[r_t = \gamma^{-t} P_i \mid s_t = s, \pi_k]$
  end for
  for all $D_i \in D_1, .., D_N$ **do**
    $R(s, s') = \gamma \hat{P}^{\pi_k}(s') - \hat{P}^{\pi_k}(s)$
    $R(s_{t_0}, s_{t_1}) = \gamma \hat{P}^{\pi_k}(s_{t_1})$
    for all (s, a) **do**
      Update the state-action value function $Q^{\pi_k}(s, a)$ with R
    end for
    Update the policy: $\pi_{k+1}(s) = \text{argmax}_a\, Q^{\pi_k}(s, a)$
  end for
until $\|\hat{P}^{\pi_k} - \hat{P}^{\pi_{k-1}}\| \leq \epsilon$

return R

$$R^{RS}(s, s') = \begin{cases} \gamma \hat{P}^\pi(s') & \text{if } s = s_{t_0} \\ \gamma \hat{P}^\pi(s') - \hat{P}^\pi(s) & \text{otherwise} \end{cases} \quad (1)$$

Let π be the last policy computed during reward shaping. Given the reward function R returned by the algorithm, the returns for a dialogue ending at a terminal state s_{t_f} are as in Equation 2.

$$\begin{aligned} r_{t_0} &= \gamma^{t_f} \hat{P}^{\pi}(s_{t_f}) \\ r_t &= \gamma^{t_f - t} \hat{P}^{\pi}(s_{t_f}) - \hat{P}^{\pi}(s_t) \end{aligned} \quad (2)$$

The idea behind reward shaping is to distribute the estimated performance score for a given dialogue among the decisions taken by the dialogue manager during this dialogue. In many cases, when an evaluator of system performance is built, the evaluation $\hat{P}$ is distributed as a reward to the system at the end of the dialogue (Walker et al. 1997a) and the return at time t is $r_t = \gamma^{t_f - t} \hat{P}$. It was shown that the reward function computed by reward shaping could lead to faster learning (El Asri, Laroche, and Pietquin 2013).

Task Completion Transfer Learning

Algorithm 2 describes the off-line computation done by TCTL.

Algorithm 2 Task Completion Transfer Learning

Require:

- A set of evaluated dialogues $\mathcal{D} = D_1, ..., D_N$ with numerical performance scores $P_1, ..., P_N$
- An MDP$\backslash R$ $M = (S, A, T, \gamma)$

Separate $\mathcal{D}$ into two partitions: dialogues with task completion ($\mathcal{D}^+$) and without task completion ($\mathcal{D}^-$)
Initialise $\dot{S} = S -$ {final states}

for all $D_i \in \mathcal{D}^+$ **do**
  for all $s_k \in D_i$ **do**
    if s_k is final and $\exists D_j \in \mathcal{D}^-$ such that s_k is final in D_j **then**
      $\dot{S} = \dot{S} \cup \{s_k^+, s_k^-\}$
    else
      $\dot{S} = \dot{S} \cup \{s_k\}$
    end if
  end for
end for

Compute $\dot{R}$ by running reward shaping on $\dot{M} = (\dot{S}, A, T, \gamma)$
Compute an optimal policy $\dot{\pi}$ and its corresponding value function $Q^{\dot{\pi}}$ for $\dot{M}$ with batch learning on $\mathcal{D}$
Compute R^{RS} by running reward shaping on M
Compute an optimal policy π^{RS} and its corresponding value function $Q^{\pi^{RS}}$ for M with batch learning on $\mathcal{D}$
return $R^f : (s_t, a_t, s_{t+1}) \mapsto R^{RS}(s_t, s_{t+1}) + Q^{\dot{\pi}}(s_t, a_t) - Q^{\pi^{RS}}(s_t, a_t)$

Let $M = (S, A, T, \gamma)$ be the MDP modelling dialogue management. Let $\mathcal{D} = (D_i)_{i \in 1..N}$ be a set of evaluated dialogues with the SDS and $(P_i)_{i \in 1..N}$ be their performance scores. Two partitions are formed from this set of dialogues: one partition $\mathcal{D}^+$ contains the dialogues where the task was completed and the other partition $\mathcal{D}^-$ contains the unsuccessful dialogues. The final states reached during the dialogues of these two partitions are then examined: if one state s_k is encountered in both $\mathcal{D}^+$ and $\mathcal{D}^-$, it is duplicated. This results in two states : s_k^+ is associated to task completion and s_k^- is associated to task failure. The state space with the duplicated final states is called $\dot{S}$ and the corresponding $MDP\backslash R$ is $\dot{M}$.

Then, we apply reward shaping to $\dot{M}$ and associate the resulting reward function $\dot{R}$ with a batch RL algorithm such as Least-Squares Policy Iteration (LSPI, (Lagoudakis and Parr 2003)) in order to learn a policy from $\mathcal{D}$. We call this policy $\dot{\pi}$. Likewise, we run reward shaping on M to deduce a reward function R^{RS}.

Having task completion embedded as a feature of a dialogue's final state enables to use $\dot{\pi}$ as a starting policy for online learning with M. Indeed, the only difference between M and $\dot{M}$ concerns their sets of final states so a policy learnt with $\dot{M}$ can be reused with M. Nevertheless, since the reward functions $\dot{R}$ and R^{RS} were learnt on different state spaces, the $Q-$function associated to $\dot{\pi}$ cannot serve as initial value for M. Transfer learning from $\dot{M}$ to M is done by adding an adjustment term to R^{RS}.

For a given dialogue of $\mathcal{D}$, s_f is the final state reached in S and $\dot{s}_f$ the one reached in $\dot{S}$. Let $(s, a) \in S \times A$ be a state-action couple visited during the dialogue and let t_s be the dialogue turn when a was decided to be taken at state s. According to equation 1,

$$\begin{aligned} \dot{r}_{t_s} &= \sum_{t_k=t_s}^{t_f} \gamma^{t_k - t_s} \dot{R}(s_k, s_{k+1}) \\ &= \sum_{t_k=t_s}^{t_f} \gamma^{t_k - t_s} (\gamma \hat{P}^{\dot{\pi}}(s_{k+1}) - \hat{P}^{\dot{\pi}}(s_k)) \\ &= \gamma^{t^f - t_s} \hat{P}^{\dot{\pi}}(\dot{s}_f) - \hat{P}^{\dot{\pi}}(s) \end{aligned}$$

$$\begin{aligned} r_{t_s}^{RS} &= \sum_{t_k=t_s}^{t_f} \gamma^{t_k - t_s} R^{RS}(s_k, s_{k+1}) \\ &= \gamma^{t^f - t_s} \hat{P}^{\pi^{RS}}(s_f) - \hat{P}^{\pi^{RS}}(s) \end{aligned}$$

$$\begin{aligned} \dot{r}_{t_s} - r_{t_s}^{RS} &= \gamma^{t^f - t_s} \hat{P}^{\dot{\pi}}(\dot{s}_f) - \hat{P}^{\dot{\pi}}(s) \\ &\quad -\gamma^{t^f - t_s} \hat{P}^{\pi^{RS}}(s_f) - \hat{P}^{\pi^{RS}}(s) \\ &= \gamma^{t^f - t_s} (\hat{P}^{\dot{\pi}}(\dot{s}_f) - \hat{P}^{\pi^{RS}}(s_f)) \\ &\quad -(\hat{P}^{\dot{\pi}}(s) - \hat{P}^{\pi^{RS}}(s)) \end{aligned} \quad (3)$$

$\gamma^{t^f - t_s}(\hat{P}^{\dot{\pi}}(\dot{s}_f) - \hat{P}^{\pi^{RS}}(s_f))$ is the difference of performance estimation between $\dot{\pi}$ and π^{RS} for the dialogues ending with state s_f or $\dot{s}_f$, depending on the considered state space. This term is thus an indicator of the non-observability of task completion and $Q^{\dot{\pi}}(s, a) - Q^{\pi^{RS}}(s, a) + (\hat{P}^{\dot{\pi}}(s) - \hat{P}^{\pi^{RS}}(s)) =$

$E\left[\gamma^{t^f - t_s}(\hat{P}^{\dot{\pi}}(\dot{s}_f) - \hat{P}^{\pi^{RS}}(s_f)) \mid s_0 = s, a_0 = a\right]$ averages this non-observability, starting from the couple (s, a), over the whole corpus. Note that it is possible to factorise by $\gamma^{t^f - t_s}$ because $Q^{\pi^{RS}}(s, a)$ and $Q^{\dot{\pi}}(s, a)$ are computed on the same corpus, only the nature of the final states changes from one computation to the other. $Q^{\dot{\pi}}(s, a) - Q^{\pi^{RS}}(s, a)$ is used to adjust the performance estimation done by R^{RS}. For each transition (s, a, s'), we add to $R^{RS}(s, s')$ a coefficient correcting the performance estimation made by R^{RS} via the difference of $Q-$values between π^{RS} and $\dot{\pi}$ for the taken action a. The information relative to the non-observability of task completion is thus embedded in the actions. We note $\hat{Q}^{\dot{\pi}}$ and $\hat{Q}^{\pi^{RS}}$ the Q-functions estimated on $\mathcal{D}$ with respectively $\dot{R}$ and R^{RS}.

The reward function R^f returned by TCTL is defined as follows: $\forall(s, a, s')$,

$$\begin{aligned} & R^f(s, a, s') \\ = \; & R^{RS}(s, s') + \hat{Q}^{\dot{\pi}}(s, a) - \hat{Q}^{\pi^{RS}}(s, a) \end{aligned} \quad (4)$$

Experimental validation

Protocol

We applied TCTL to a restaurant-seeking dialogue system similar to the one presented in (Lemon et al. 2006b). During the interaction, the system needs to fill three slots to provide the information to the user: type of food, price range and city area. Instead of keeping in memory the values given for each slot (*e.g.* type of food = Italian, price range = cheap, city area = dowtown) and a probability for each value to have been correctly understood , the system only remembers if a slot is empty, filled or confirmed. The actions the system can perform are the following: greet the user, ask a slot, explicitly confirm a slot, implicitly confirm a slot and close the dialogue.

We simulated 1500 scenario-based dialogues with this system. User behaviour was simulated with a Bayesian network, as proposed in (Pietquin, Rossignol, and Ianotto 2009). The parameters of the Bayesian network were learned on the 1234 human-machine dialogues described in (Williams and Young 2007). System policy was uniform in order to gather as many observations as possible for each state. For each dialogue, the values of the slots understood by the system were compared to the ones intended by the simulated user. Each dialogue was scored according to the function in equation 5, where nbRight is the number of slots which were correctly filled, nbWrong the number of slots incorrectly filled, nbEmpty, the number of slots left empty and nbTurns the number of dialogue turns.

$$\begin{aligned} \text{score} = & -3 \times \text{nbEmpty} + 0.25 \times \text{nbRight} \\ & - 0.75 \times \text{nbWrong} - 0.015 \times \text{nbTurns} \end{aligned} \quad (5)$$

We trained TCTL on 26 training sets of $50, 100, 150, ..., 1300$ randomly drawn dialogues and tested it on 26 test sets with the remaining $1450, 1400, 1350, ...200$ dialogues. On each training set, we computed $\dot{R}$, R^{RS} and R^f. On each test set, we compared the returns at time 0 according to these functions to the simulated performance scores computed with Equation 5. We repeated this process 150 times, each time drawing different training sets.

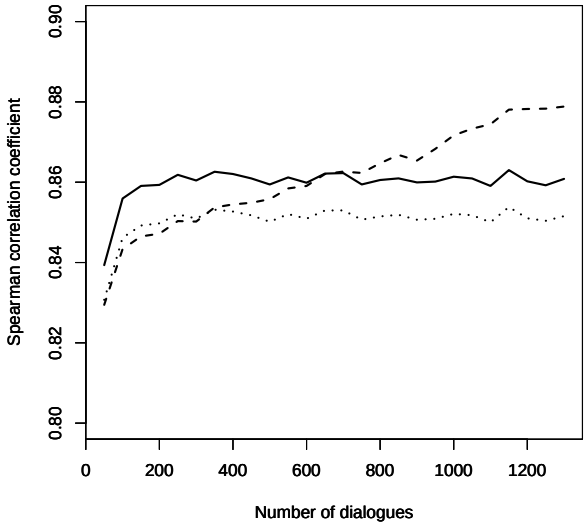

Figure 1: Spearman correlation coefficient between the simulated returns and the ones computed with $\dot{R}$ (plain line), R^{RS} (dotted line) and R^f (dashed line).

Results

Figure 1 displays the Spearman correlation coefficients between the performance scores computed with Equation 5 and the returns given by $\dot{R}$, R^{RS} and R^f on the 26 test sets, averaged on the 150 runs. The Spearman correlation coefficient (Spearman 1904) compares the rankings of the dialogues, if two rankings are the same, the coefficient is equal to 1. If they are opposite, the coefficient is equal to -1. In order to learn a policy similar to the one that would be learnt with the scoring function, the inferred reward function should rank the dialogues in a similar way.

In Figure 1, it can be seen that a training set of 200 dialogues is enough for $\dot{R}$ and R^{RS} to reach their highest correlation value (respectively 0.86 and 0.85). As expected, the correlation with $\dot{R}$ is always higher than the one with R^{RS}. As for R^f, it keeps improving its quality as a performance estimator and even surpasses $\dot{R}$ for training sets bigger than 700 dialogues to reach a correlation of 0.87 for a training set of 1300 dialogues. Besides, Figure 1 shows that R^f is a better estimator of performance than R^{RS} for any training set of size higher than 300 dialogues. 300 dialogues corresponded to the training set necessary to learn an optimal policy (which consisted of asking for and then confirming each slot turn by turn) with $\dot{R}$.

Discussion

For a small evaluated corpus of dialogues (under 300 in the previous example), the quality of R^f is similar to the one of R^{RS}. Then, once the training set is large enough to learn

an optimal policy with $\dot{R}$, the adjustment term in R^f helps predict better the performance of each dialogue than R^{RS}.

Another interesting result is that, on training sets bigger than 700 dialogues, R^f gives a more accurate ranking of dialogues than $\dot{R}$. This can be explained by the fact that, in this case, the task was completed if and only if every slot had been correctly understood by the system. $\dot{R}$ was thus learnt on a state space that only separated two cases of task completion. Therefore, $\dot{R}$ could not easily distinguish the cases were the task had been partially completed (e.g. one slot correctly understood and the other two misunderstood). However the adjustment term in R^f takes into account the actions taken during the dialogue and this helped to rank correctly these dialogues. Indeed, in these dialogues, there are many user time outs or speech recognition rejections that make the dialogue stagnate at the same state and R^f is more precise than $\dot{R}$ because it penalises these outcomes. These results show that not only TCTL enables to provide a better estimator of dialogue performance than the one inferred by reward shaping, it also learns an accurate estimator of the cases where task completion can have more than two values.

We could have included partial task completion in $\dot{S}$ duplicating final states into four states (corresponding to 0 to 3 correctly filled states) to make task completion fully observable in $\dot{M}$. Nevertheless, our goal is to use reward shaping with dialogues scored by users and in this case, the parameters explaining the scores cannot be entirely known. We simulated here the fact that users will score dialogues according to partial task completion and we showed that in this situation, TCTL can be successfully used to estimate dialogue performance.

It is thus possible to optimise the exploitation of an evaluated corpus to deal with online non-observability of features as crucial as task completion. We believe TCTL is a promising strategy for building a reward function which can be used online. In future work, we will compare the performance of the policy learnt with R^f to the one learnt with R^{RS} on other simulated dialogues to confirm that R^f entails a policy that leads to higher performance.

As said in the introduction, another non-observable yet paramount feature to predict user satisfaction is the number of speech recognition errors. We intend to extend our methodology for computing a reward function to handling this feature, adapting work from (Litman and Pan 2002) who showed it is possible to build a model to predict the amount of errors from a set of annotated dialogues. If such a model can be built efficiently, it will then be possible to integrate speech recognition errors to the set of features composing the state space.

Another direction for future work is to additionally exploit a set of evaluated dialogues to build jointly the state space and the reward function of a given SDS.

Conclusion

This paper has introduced Task Completion Transfer Learning (TCTL) for learning Spoken Dialogue Systems (SDS). When task completion is non-observable online, TCTL transfers the knowledge of task completion on a corpus of evaluated dialogues to online learning. This is done by computing a reward function embedding information about task completion.

TCTL was tested on a set of scenario-based simulated dialogues with an information providing system. The reward function computed by TCTL was compared to the one computed with a reward inference algorithm. It was shown that for a set of dialogues sufficiently large, TCTL returns a better estimator of dialogue performance.

Future work will focus on including speech recognition errors to the model and optimising the conception of the SDS state space.

Acknowledgements

The authors would like to thank Heriot-Watt University's *Interaction lab* for providing help with DIPPER.

References

Chandramohan, S.; Geist, M.; Lefèvre, F.; and Pietquin, O. 2011. User simulation in dialogue systems using inverse reinforcement learning. In *Proc. of Interspeech*.

Cohen, J. 1960. A coefficient of agreement for nominal scales. *Educational and Psychological Measurement* 20:37–46.

El Asri, L.; Lemonnier, R.; Laroche, R.; Pietquin, O.; and Khouzaimi, H. 2014. NASTIA: Negotiating Appointment Setting Interface. In *Proc. of LREC (to be published)*.

El Asri, L.; Laroche, R.; and Pietquin, O. 2012. Reward function learning for dialogue management. In *Proc. of STAIRS*.

El Asri, L.; Laroche, R.; and Pietquin, O. 2013. Reward shaping for statistical optimisation of dialogue management. In *Proc. of SLSP*.

Lagoudakis, M. G., and Parr, R. 2003. Least-squares policy iteration. *Journal of Machine Learning Research* 4:1107–1149.

Laroche, R.; Putois, G.; Bretier, P.; Aranguren, M.; Velkovska, J.; Hastie, H.; Keizer, S.; Yu, K.; Jurčíček, F.; Lemon, O.; and Young, S. 2011. D6.4: Final evaluation of classic towninfo and appointment scheduling systems. Technical report, CLASSIC Project.

Larsen, L. B. 2003. Issues in the evaluation of spoken dialogue systems using objective and subjective measures. In *Proc. of IEEE ASRU*, 209–214.

Lemon, O., and Pietquin, O. 2012. *Data-Driven Methods for Adaptive Spoken Dialogue Systems*. Springer.

Lemon, O.; Georgila, K.; Henderson, J.; and Stuttle, M. 2006a. An ISU dialogue system exhibiting reinforcement learning of dialogue policies: Generic slot-filling in the talk in-car system. In *Proc. of EACL*.

Lemon, O.; Georgila, K.; Henderson, J.; and Stuttle, M. 2006b. An ISU dialogue system exhibiting reinforcement learning of dialogue policies: generic slot-filling in the TALK in-car system. In *Proc. of EACL*.

Litman, D. J., and Pan, S. 2002. Designing and evaluating an adaptive spoken dialogue system. *User Modeling and User-Adapted Interaction* 12:111–137.

Möller, S.; Krebber, J.; Raake, E.; Smeele, P.; Rajman, M.; Melichar, M.; Pallotta, V.; Tsakou, G.; Kladis, B.; Vovos, A.; Hoonhout, J.; Schuchardt, D.; Fakotakis, N.; Ganchev, T.; and Potamitis, I. 2004. INSPIRE: Evaluation of a Smart-Home System for Infotainment Management and Device Control. In *Proc. of LREC*.

Pietquin, O.; Rossignol, S.; and Ianotto, M. 2009. Training Bayesian networks for realistic man-machine spoken dialogue simulation. In *Proc. of IWSDS 2009*.

Raux, A.; Langner, B.; Black, A.; and Eskenazi, M. 2003. LET'S GO: Improving Spoken Dialog Systems for the Elderly and Non-natives. In *Proc. of Eurospeech*.

Rieser, V., and Lemon, O. 2011. Learning and evaluation of dialogue strategies for new applications: Empirical methods for optimization from small data sets. *Computational Linguistics* 37.

Singh, S.; Kearns, M.; Litman, D.; and Walker, M. 1999. Reinforcement learning for spoken dialogue systems. In *Proc. of NIPS*.

Spearman, C. 1904. The proof and measurement of association between two things. *American Journal of Psychology* 15:72–101.

Sugiyama, H.; Meguro, T.; and Minami, Y. 2012. Preference-learning based Inverse Reinforcement Learning for Dialog Control. In *Proc. of Interspeech*.

Taylor, M. E., and Stone, P. 2009. Transfer learning for reinforcement learning domains: A survey. *Journal of Machine Learning Research* 10:1633–1685.

Walker, M.; Hindle, D.; Fromer, J.; Fabbrizio, G.; and Mestel, C. 1997a. Evaluating competing agent strategies for a voice e-mail agent. In *Proc. of EuroSpeech*.

Walker, M. A.; Litman, D. J.; Kamm, C. A.; and Abella, A. 1997b. PARADISE: a framework for evaluating spoken dialogue agents. In *Proc. of EACL*, 271–280.

Walker, M. A.; Fromer, J. C.; and Narayanan, S. 1998. Learning optimal dialogue strategies: A case study of a spoken dialogue agent for email. In *Proc. of COLING/ACL*, 1345–1352.

Williams, J. D., and Young, S. 2007. Partially observable markov decision processes for spoken dialog systems. *Computer Speech and Language* 21:231–422.