Transfer Learning for User Adaptation in Spoken Dialogue Systems

Aude Genevay
Orange Labs
Issy les Moulineaux, France
aude.genevay@gmail.com

Romain Laroche
Orange Labs
Issy les Moulineaux, France
romain.laroche@orange.com

ABSTRACT

This paper focuses on user adaptation in Spoken Dialogue Systems. It is considered that the system has already been optimised with Reinforcement Learning methods for a set of users. The goal is to use and transfer this prior knowledge to adapt the system to a new user as quickly as possible without impacting asymptotic performance. The first contribution is a source selection method using a multi-armed stochastic bandit algorithm in order to improve the jumpstart, *i.e.* the average performance at the start of the learning curve. Contrarily to previous source selection methods, there is no need to define a metric between users, and it is parameter free. The second contribution is an innovative method for selecting the most informative transitions within the previously selected source, to improve the target model, in such a way that only transitions that were not observed with the target user are transferred from the selected source. For our experimentation, Reinforcement Learning is performed with the Fitted Q-Iteration algorithm. Both methods are validated on a negotiation game: an appointment scheduling simulator that allows the definition of simulated user models adopting diversified behaviours. Compared to state-of-the-art transfer algorithms, results show significant improvements for both jumpstart and asymptotic performance.

General Terms

Algorithms

Keywords

Reinforcement Learning, Transfer Learning, Markov Decision Processes, Online Learning, User Profiling, Multi-Armed Bandit, Jumpstart, Asymptotic Performance Spoken Dialogue Systems

1. INTRODUCTION

Spoken dialogue systems have the ability to interact directly with a human through speech. Reinforcement Learning [35] is a popular framework for dialogue management in spoken dialogue systems [28, 34]. This allows the system to learn its optimal behaviour by exploring different options and getting a delayed reward denoting the performance of its actions. Such an approach is generally solved by casting the system into a Markov Decision Process (MDP). While previous work in spoken dialogue systems mostly focused on learning a policy which is good for all users in average [27, 39], the goal here is to learn a personalised policy which is adapted to the user's characteristics.

The general goal in this paper is to transfer in an efficient way relevant information from well-known[1] users called *sources* to a new user called *target*. This problem, known for a decade as Transfer Learning [7, 31], and adapted to Reinforcement Learning more recently [38, 25] faces a trade-off between a good *jumpstart* (an efficient initial policy for a new user) and a good *asymptotic performance* (average return received after infinitely many iterations). Learning an average policy for all users and reusing it for a new similar user provides a good jumpstart but leads to a poor asymptotic performance, while learning from scratch for a new user has the opposite effect. The difficult task is therefore to imprint the virtuous bias from the sources into the learning on the target, and at the same time erase the vicious bias.

In order to optimise this trade-off, two issues are tackled in this article. The first one is choosing relevant sources for transfer. More precisely, which users, trajectories or inferred knowledge are similar or informative for the target to improve the learning procedure. The second issue is how to transfer knowledge to the target. Once relevant sources have been identified, the goal is to use this supplementary information, and merge it wisely with the new target information. Although research on Transfer Learning has been quite dynamic in recent years, these ideas have mostly been ignored in dialogue literature until 2013.

In the dialogue literature, Gašić [16, 17] uses Gaussian Processes Reinforcement Learning [11] for Transfer Learning cast as Partially Observable Markov Decision Processes for domain extension in spoken dialogue systems. The system has been optimised on a source domain, and they want to use this knowledge on a new target domain. As a consequence, there is no source selection, contrarily to [5] which uses the same framework for user adaptation of dialogue systems to different classes of dysartric users. However, their approach requires defining a metric between users for source selection. It works well in a setting like theirs where the user profiles are defined by a similarity based on the voice signature, which can be easily defined and measured online. But on a dialogue management level, with generic user behaviour

Appears in: *Proceedings of the 15th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2016), J. Thangarajah, K. Tuyls, C. Jonker, S. Marsella (eds.), May 9–13, 2016, Singapore.*

[1] By well-known, we mean that these users have interacted sufficiently with the system so that we can consider that it has converged to an optimal policy.

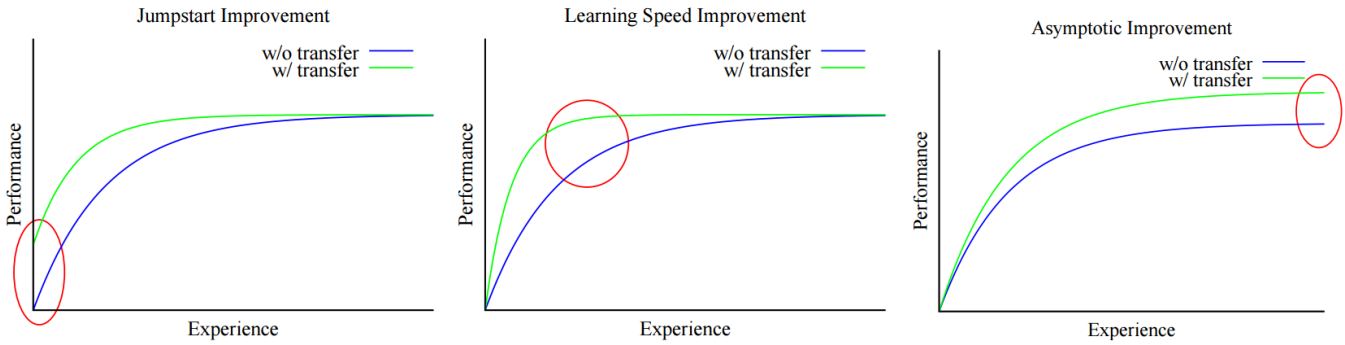

Figure 1: The three objectives of Transfer Learning.

profiles, it is hard to design and to measure: the observed features that are necessary to compute the similarity may be biased by the policy followed during data collection.

The first contribution of this article is a novel source selection method casting the source selection problem as a multi-armed bandit problem. Instead of searching for the most similar source, the paradigm is changed to looking for a model learnt from a source user that performs well with the target user. Since the objective is to minimise the regret at the beginning of the learning curve, it should induce an improvement in the jumpstart. Empirical results confirm this but also shows that the source selection is improved.

The transfer process consists in an instance transfer: the transitions from one Markov Decision Process state to another, after performing an action. Transitions from the trajectories of the chosen source are added to those already collected from the target as suggested by Lazaric [26]. Reinforcement Learning is then performed with a batch reinforcement learning algorithm. The second contribution consists in selecting transitions from the selected source with an innovative method which only keeps the ones embedding information missing from the target trajectories. Again, the empirical results show an asymptotic performance improvement against two state-of-the-art transfer algorithms.

Thus, this paper focuses on the individual adaptation of a system to its specific users. Any kind of personal assistant is a practical application, like in a connected home, where each house has its own configuration, inside which each house member might have different habits, implying different exploitable prior knowledge, and different ways to express themselves, which requires adapted behaviours from the system. User adaptation is also useful in negotiation-based dialogues [32, 19], which is the experimental setting that has been chosen in this paper. Negotiation dialogue has been lately solved by abandoning the single-agent reinforcement learning paradigm in favour of multi-agent reinforcement learning [18] and stochastic games [2]. However, considering that most dialogue tasks, including negotiation ones such as appointment scheduling, are cooperative, the user adaptation approach makes sense to yield maximum rewards.

The empirical studies in this article are led on a recently published negotiation dialogue game abstraction [23]. The dialogue negotiation literature[2] has been abstracted as an agreement problem over a shared set of options. The goal for the players is to reach an agreement and select an option. This negotiation dialogue game can be parametrised to make it zero-sum, purely cooperative, or general sum. In our experiments, it is set to a purely cooperative game and it is exposed as an appointment scheduling problem.

The transfer framework, method and contributions are detailed from end to end in the next section. Then, the negotiation game designed for our experiments is thoroughly introduced. Afterwards, the settings of the experiment and the results are presented. Eventually, we conclude the paper by proposing some areas of improvements.

2. FRAMEWORK AND ALGORITHMS

This section presents the framework and algorithms for transfer between users in a Reinforcement Learning domain. First, it presents the theoretical framework for Transfer in Reinforcement Learning that is used in this article, which is largely inspired from [25]. Figure 1 recalls the three objectives of Transfer Learning [22]: jumpstart or the improvement of the starting model, learning speed, and asymptotic performance. The two contributions of this articles concern: jumpstart thanks to a multi-armed stochastic bandit source selection approach, and asymptotic performance[3] improvement thanks to a density-based transition selection within the previously selected source.

2.1 Transfer in Reinforcement Learning

The dialogue system is formalised as a Markov Decision Process (MDP) which is the classical framework for Reinforcement Learning. An MDP is a tuple $\langle \mathcal{S}, \mathcal{A}, P, R, \gamma \rangle$ where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $P : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$ is the Markovian transition stochastic function, $R : \mathcal{S} \to \mathbb{R}$ is the immediate reward stochastic function, and γ is the discount factor.

A *trajectory* $\langle s_t, a_t, s_{t+1}, R_t \rangle_{t \in [\![0,T-1]\!]}$ is the projection into the MDP of the task episode, called *dialogue* in our domain. The goal is to generate trajectories with high discounted cumulative reward, also called more succinctly *return*:

$$r(\langle s_t, a_t, s_{t+1}, R_t \rangle_{t \in [\![0,T-1]\!]}) = \sum_{t=0}^{T-1} \gamma^t R_t \qquad (1)$$

To do so, one needs to find a policy $\pi : \mathcal{S} \to \mathcal{A}$ which yields optimal expected returns. Formally, this means find-

[2] [12] considers sets of furniture, [8, 18] resource trading, and [24, 10] appointment scheduling.

[3] against other transfer algorithms. Against learning without transfer, learning speed is improved.

ing the policy which maximises the following Q-function:

$$Q^*(s,a) = Q^{\pi^*}(s,a) = \underset{\pi}{\operatorname{argmax}}\, \mathbb{E}_\pi \left[\sum_{t \geq 0} \gamma^t R_t | s, a \right] \quad (2)$$

A Transfer Learning algorithm intends to make the learning for a new *task* easier, by using antecedent knowledge from one or several similar tasks. Let $\mathscr{U} = \{\mathscr{p}_u^k\}_{k \in [\![1,n]\!]}$ be the set of K different users defining as many tasks M_u^k, $\mathscr{M} = \{M_u^k\}_{\mathscr{p}_u^k \in \mathscr{U}}$ be the space of tasks, $\mathscr{K}$ be the space of knowledge transferred from one task to another, $\mathscr{H}$ be the space of hypotheses for solving the tasks, and $\mathscr{L} : \mathscr{K} \to \mathscr{H}$ be the general learning algorithm.

In our setting, a task M_u^k is defined as an MDP. The difference between a task and another is solely dependent on the users defining the task. As they are part of the environment, the state space $\mathscr{S}_u^{\mathscr{k}}$, the action space $\mathscr{A}_u^{\mathscr{k}}$ and the discount factor γ_u^k are invariant and $M_u^k = \langle \mathscr{S}, \mathscr{A}, P_u^k, R_u^k, \gamma \rangle$.

Representation transfer [37] is an active research area: one can find studies on option based transfers [20, 30], which are dependant on hierarchical reinforcement learning [36], on Proto-value functions [29, 15], or on representation transfer for Gaussian Processes Reinforcement Learning [16, 17, 5]. This article does not deal with representation transfer, and has chosen instead to focus on transfer selection: source selection [14, 30] and instance selection [26].

As a consequence, in the remainder of the article, $\mathscr{L}$ can be instantiated with any batch learning algorithm taking as an input a set of transitions (or samples, or instances) $\langle s, a, s', r \rangle \in \mathscr{S} \times \mathscr{A} \times \mathscr{S} \times \mathbb{R}$ within a trajectory, s being the current state, a the taken action, s' the next reached state and r the immediate reward. Therefore, $\mathscr{K} = (\mathscr{S} \times \mathscr{A} \times \mathscr{S} \times \mathbb{R})^N$, where N is the number of transferred transitions.

2.2 Similarity-based source selection

The similarity-based source selection is widely spread in the Transfer Learning literature [4, 38, 5]. In [5], four dysartric speaker profiles are defined. Then, the goal is to transfer the Gaussian Processes Reinforcement Learning [11] performed on those profiles to the new target user. A similarity between users is defined in order to identify the profile that is the closest to the speaking profile of the target. This similarity is based on statistics on the audio signal features from their voices. When faced with a new user, the closest user is defined as :

$$\mathscr{p}_u^* = \underset{\mathscr{p}_u^k \in \mathscr{U}}{\operatorname{argmin}} ||\hat{\nu}_u^k - \hat{\nu}_u^{new}|| \quad (3)$$

where $||.||$ is a norm, $\hat{\nu}_u^k$ and $\hat{\nu}_u^{new}$ are the vectors of statistics for user $\mathscr{p}_u^k$ and the target user respectively. Unfortunately, this approach has several shortcomings.

Firstly, it is difficult in a general setting to identify the insightful statistics for selecting the best user. It is even more complex to define the $||.||$ norm. In [5], on a specific user adaptation problem, three different metrics are tested and yield heterogeneous results.

Secondly, for behavioural profiling, those insightful statistics often require knowledge that cannot be obtained directly: for instance, in order to know the sentence error rate of the speech recognition, the system needs to explicit its understanding, this statistic collection is therefore dependent on the followed policy, and it is a source of bias in the statistics.

Thirdly, when no trajectory has been collected on the target, a generic, default policy learnt uniformly over all users is used to gather the first n_{first} dialogues. This policy experiences to be more efficient than a random policy, but still yields a large regret.

2.3 Performance-based source selection

The first contribution of this article is a statistics independent source selection: instead of selecting a similar source user, the approach consists in selecting an efficient source user in a spirit close to Policy Library through Policy Reuse [14]. The source selection is modelled as a multi-armed stochastic bandit problem, where each arm corresponds to the optimal policy π_s^k learnt with a source user $\mathscr{p}_u^k$; where *pulling an arm* a_k means using this policy π_s^k for a whole dialogue, and thus generate a whole trajectory; and where the bandit reward for the chosen arm corresponds to the dialogue return defined by Equation 1. In practice, such a bandit algorithm can learn to select an efficient policy to use and to transfer from with only a few dialogue returns, or equivalently arm pulls in the bandit setting. There is no guarantee that the selected arm will be the best, but with high probability, it corresponds an efficient policy, which is sufficient.

In order to select the right bandit algorithm, it is important to define exactly what is the main objective of the algorithm. Is it to select the best arm with the highest probability as in action elimination algorithms [13]? Is it to minimise the regret as in Upper Confidence Bound (UCB) algorithms [1]? The answer is that there is no need to select the best arm, only a good enough arm is required to be used as a source, but it is important to guarantee a good jumpstart for the Transfer Learning algorithm while the first trajectories are collected from the target, and this is exactly what UCB does by minimising the regret.

More specifically, as the time scale for the jumpstart improvement is very short, a variance estimation is impossible. As a consequence, the simplest UCB algorithm: the UCB1 algorithm [1] has been implemented. First, it pulls each arm once, then it pulls the arm a^k that maximises the following equation:

$$\hat{\mu}^k + \rho \sqrt{\frac{\ln n}{n_k}} \quad (4)$$

where $\hat{\mu}^k$ is the empirical average return after pulling arm a^k, where ρ is a constant parameter controlling the magnitude of exploration, where n is the total number of times UCB pulled arms, and where n_k is the number of times arm a_k has been pulled.

The problem becomes non-stationary when the arms change, *i.e.* if the user policies evolve. Fortunately, the source selection intervenes during the jumpstart. As a consequence, the UCB algorithm is only used during this first stationary step of length n_{first}. After that, we are able to select a user with an efficient policy to get source from and to learn a target policy that is better than any of the source ones.

2.4 Transition selection: baseline

The transfer algorithm uses trajectories from the chosen source to initialise the batch Reinforcement Learning algorithm, and trajectories collected with the new user are added gradually to the batch. The previous subsection explained the way the first step is handled, when no target trajectory

has been collected. This subsection describes how the transitions issued from source and target trajectories are merged.

The classical way to do it consists in initialising $K \in \mathscr{K}$ with $n_{transfer}$ transitions transferred with the selected source, and to add the target transitions as the trajectories are collected [38, 25]. The target transitions will eventually overwhelm the source ones. One might be tempted to eventually remove the source transitions from K. It is reported that it is often more efficient to keep them in. The reason is that the source is used to provide a prior on the similarity between the tasks. Removing the source transitions leaves ignorance on the performance of some state-action couples, ignorance that has to be made up for. Both baselines with and without replacement will be considered.

2.5 Density-based transition selection

When a source is selected, it is crucial to add the right amount of information to the initial batch: too little will result in a bad jumpstart, while too much will lead to overfitting the source. It is also crucial to add the right type of information. It is pointless to transfer source transitions that are frequent in the target trajectories: this leads to keeping a bias from the source. On the contrary, the transitions that are absent from the target trajectories are mostly state-actions couples that yielded poor performance in the source and that the target would be better off not exploring. These absent transitions should be chosen so that the state-action space is well covered.

To solve this issue, a novel algorithm is introduced to select relevant transitions within the trajectories of the previously selected source. Given a transition from the source $\langle s, a, s', r\rangle$, all the transitions already in the batch which contain action a are considered. If there is a transition $\langle s_i, a, s'_i, r_i\rangle$ such that $||s - s_i||_2 \leq \eta$ then the transition is not added to the batch. This means that only transitions which are far enough from existing transitions are added to have just the amount of information needed. The choice of η depends on the problem and should be tuned carefully. A large value for this parameter will lead to adding too few transitions to the batch, while a small value will have the opposite effect.

The use of a similarity and a threshold is obviously a shortcoming. Although, it does not apply to the same problem as the similarity measure between reinforcement learning tasks. In practice, it is easier to define a similarity between MDP states $\mathcal{S}$, which can be computed with a simple Euclidean distance, than between user behaviours, which are determined by stochastic functions P_u^k and R_u^k and are therefore not directly observable (since the internal user's states are unknown). A wide range of algorithms for learning a state representation exists [3, 9].

2.6 End-to-end algorithm

Algorithm 1 details the pseudo-code for the transfer learning procedure. It lays out more clearly the articulation of the various parts of the procedure described above.

The algorithm is initialised with a list of known users as well as the new user for which learning has to be performed. To each user corresponds a previously learnt policy π_s^k and a transition database $\mathscr{D}^k$ recording previously collected dialogues. These variables are respectively initialised to default policy and $\emptyset$ for the new target user.

Algorithm 1 Transfer for Reinforcement Learning with UCB source selection and density-based transition selection

TransferLearning()
Initialisation()
ρ_u^* = UCBSourceSelection()
for ($i = 1$; $i < nbBatch$; i++) **do**
  $\mathscr{D}^{tr}$ = DensityBasedSelection(ρ_u^*,$\mathscr{D}^{new}$)
  π_s^{new} = updatePolicy($\mathscr{D}^{new} \cup \mathscr{D}^{tr}$)
  while $n_{new} \neq 0$ (mod $batchSize$) **do**
    dial = playDialogue(ρ_u^{new}, π_s^{new})
    $\mathscr{D}^{new} \leftarrow \mathscr{D}^{new} \cup$ dial.getTransitions()
    n_{new}++
  end while
end for
π_s^{new} = updatePolicy($\mathscr{D}^{new} \cup \mathscr{D}^{tr}$)

Initialisation()
$\mathscr{U} = \left[\rho_u^1, \rho_u^2, \ldots, \rho_u^K\right]$
$\forall k \in [\![1, K]\!]$, $\rho_u^k = \{\pi_s^k, \mathscr{D}^k\}$
$\rho_u^{new} = \{\pi_s^{new}, \mathscr{D}^{new} = \emptyset\}$
$n_{new} = 0$
$nbBatch$
$batchSize$

UCBSourceSelection()
while $n_{new} < n_{first}$ **do**
  idx = UCB.Choice()
  dial = playDialogue(ρ_u^{new}, π_s^{idx})
  $\mathscr{D}^{new} \leftarrow \mathscr{D}^{new} \cup$ dial.getTransitions()
  UCB.setReturn(dial)
  n_{new}++
end while
ρ_u^* = UCB.bestArm()
return ρ_u^*

DensityBasedSelection(ρ_u^k,$\mathscr{D}^{new}$)
$\mathscr{D}^{tr} = \emptyset$
for $\langle s_i^k, a_i^k, s'^k_i, r_i^k\rangle \in \mathscr{D}^k$ **do**
  addTransition = true
  for $\langle s_j^{new}, a_i^k, s'^{new}_j, r_j^{new}\rangle \in \mathscr{D}^{new}$ **do**
    if $||s_i^k - s_j^{new}||_2 \leq \eta$ **then**
      addTransition = false
      break
    end if
  end for
  if addTransition **then**
    $\mathscr{D}^{tr} \leftarrow \mathscr{D}^{tr} \cup \{\langle s_i^k, a_i^k, s'^k_i, r_i^k\rangle\}$
  end if
end for
return $\mathscr{D}^{tr}$

First, UCB-based source selection is performed with **UCBSourceSelection()** which returns the best source user ρ_u^*. Then, **DensityBasedSelection(ρ_u^*,$\mathscr{D}^{new}$)** takes this user as an input and selects interesting transitions from $\mathscr{D}^*$ to return them in a new matrix $\mathscr{D}^{tr}$. This matrix which is further added to $\mathscr{D}^{new}$ in order to start the learning procedure.

The current policy is updated with **updatePolicy()** which implements a batch Reinforcement Learning algorithm taking a transitions matrix as an input. New dialogues are collected with this policy until the chosen batch size is reached after which the policy is updated again to collect new dialogues and so on. This update/collection process is repeated $nbBatch$ times before proceeding to the final policy update.

3. A NEGOTIATION GAME

3.1 Appointment scheduling simulator

Our experiments are run on a negotiation game designed specifically to have different user strategies [23]. It can be interpreted as an appointment scheduling problem between two players. One of the players is a user $\mathcal{p}_u^i$, and the other one is the system $\mathcal{p}_s^i$ to be optimised against/with $\mathcal{p}_u^i$. $n = 30$ time-slots are considered, and for each time-slot τ, each player $\mathcal{p}^i$ has a utility $\omega_\tau^i \in [0, 5]$ for booking it. Cooperation between players is introduced, which results in the following definition of player $\mathcal{p}^i$'s immediate reward at the end of the dialogue:

$$R^i(s_T) = \omega_\tau^i + \alpha\, \omega_\tau^j \tag{5}$$

where s_T is the last state reached at the end of the dialogue, where τ is the chosen time-slot and where $\alpha \in \mathbb{R}$ is the cooperation parameter. If both players cannot agree after n turns (which would correspond to a strategy in which each player enumerates his time-slots in descending order of preference one after the other) the final immediate rewards $R^i(s_T)$ and $R^j(s_T)$ are equal to 0 for both players. As well, if the system misunderstands and agrees on a time-slot which isn't the one proposed by the user, $R^i(s_T) = R^j(s_T) = 0$.

A system $\mathcal{p}_s$ and a user $\mathcal{p}_u$ act each one in turn, starting randomly by one or the other. They have four possible parametric actions:

- Accept(τ) means that the user accepts the time-slot τ[4]. This act ends the dialogue.
- RefProp(τ) means that the user refuses the proposed time-slot and proposes instead time-slot τ.
- Repeat means that the player asks the other player to repeat his proposition.
- EndDial denotes the fact that the player does not want to negotiate anymore.

Understanding through speech recognition of system $\mathcal{p}_s$ is assumed to be noisy with a sentence error rate SER_s^k after listening to a user $\mathcal{p}_u^k$: with probability SER_s^k, an error is made, and the system understands a random time-slot instead of the one that was actually pronounced. In order to reflect human-machine dialogue reality, a simulated user always understands what the system says: $SER_u^s = 0$. We adopt the way [21] generates speech recognition confidence scores:

$$score_{reco} = \frac{1}{1 + e^{-X}} \text{ where } X \sim \mathcal{N}(c, 0.2) \tag{6}$$

Given a user $\mathcal{p}_u^k$, two parameters $(c_\perp^k, c_\top^k)$ with $c_\perp^k < c_\top^k$ are defined such that if the player understood the right time-slot, $c = c_\top^k$ otherwise $c = c_\perp^k$. The further apart the normal distribution centres are, the easier it will be for the system to know if it understood the right time-slot, given the score.

[4]Independently from the fact that τ has been proposed by the other player. If it has not, this induces a null reward.

3.2 User profiles

A user interacts symmetrically with the system. The first proposal is made randomly by the user or the system, and then they alternately make proposals until an agreement is found, or the maximal number of turns is reached.

A characteristic of a user is its intelligibility by the system, parametrised by its average sentence error rate p_{SER}. Another understanding characteristic consists in varying centres $(c_\perp^i, c_\top^i)$ for the speech recognition score. For distant $(c_\perp^i, c_\top^i)$ values, the system will easily know if it understood well.

In order to add more variability in our simulated users, two handcrafted classes of users have been implemented:

- The *Deterministic User* (parameter x) Accept(τ) if and only if $\tau \in \mathcal{T}_x$, where $\mathcal{T}_x$ is the set of its x preferred time-slots. If $\tau \notin \mathcal{T}_x$, he RefProp($\tau'$), $\tau' \in \mathcal{T}_x$ being his preferred time-slot that was not proposed before. If all $\tau \in \mathcal{T}_x$ have been refused, or if the system insists by proposing the same time-slot twice, he EndDial.
- The *Random User* (parameter p) Accept(τ) any time-slot τ asked by the system, with probability p. With probability $1 - p$, he RefProp(τ') a time-slot τ' randomly. If he's asked to repeat, he'll make a new random proposition.

3.3 Reinforcement learning implementation

Least-squares Fitted Q-Iteration is used to learn the policy with a linear parametrisation of the Q-function. The following recalls the basics. The optimal Q-function Q^* is known to verify Bellman's equation:

$$Q^*(s,a) = \mathbb{E}\left[R(s,a,s') + \gamma \max_{a'} Q^*(s',a')\right] \tag{7}$$

$$\Leftrightarrow \quad Q^* = T^* Q^* \tag{8}$$

The optimal Q-function is thus the fixed point of Bellman's operator T^* and since it is a contraction Banach's theorem ensures its uniqueness. Hence, the optimal Q-function can be obtained by iteratively applying Bellman's operator. This procedure is called Value Iteration.

When the state space is continuous (or very large) it is impossible to use Value-Iteration as such. The Q-function must be parametrised. A popular choice is the linear parametrisation of the Q-function [6]:

$$\hat{Q}_a(s) = \hat{\theta}_a^\top \Phi_a(s) \tag{9}$$

where $\Phi = \{\Phi_a\}_{a \in \mathcal{A}}$ is the feature vector for linear state representation and $\theta = \{\theta_a\}_{a \in \mathcal{A}}$ is the parameter that has to be learnt. Each dimension of θ_a represents the influence of the corresponding feature in the Q-function.

The inference problem can be solved by alternately applying Bellman's operator and projecting the result back onto the space of linear functions, and iterating these two steps until convergence.

$$\theta_a^{(i+1)} = (X_a^\top X_a)^{-1} X_a^\top y_a^{(i)} \tag{10}$$

where X_a is the observation matrix, where each line is the s_j feature vector: $(X_a)_{\sigma_a(j)} = \Phi_a(s_j)$, $y_a^{(i)}$ is a vector such that $(y_a^{(i)})_{\sigma_a(j)} = r_j + \gamma \max_{a'} \theta_{a'}^{(i)} \Phi_{a'}(s'_j)$, and $\sigma_a(j)$ is the index

role	name	user characteristics type	parameter	centres	avg return w. policy π_s^i learnt against π_u^i π_s^1	π_s^2	π_s^3	π_s^4	π_s^5
source	π_u^1	deterministic	1/10	(0,0)	**4.72**	1.90	2.77	1.63	1.74
source	π_u^2	deterministic	1/10	(-5,5)	5.22	**6.13**	4.73	2.50	2.61
source	π_u^3	deterministic	1/5	(-5,5)	5.28	**6.14**	**6.14**	3.05	3.25
source	π_u^4	random	0.3	(-5,5)	3.96	4.58	4.72	**5.09**	4.96
source	π_u^5	random	0.5	(-5,5)	4.15	4.83	5.08	5.42	**5.51**
target	π_u^a	deterministic	1/5	(-1,1)	5.12	4.75	**5.38**	2.72	2.70
target	π_u^b	deterministic	1/5	(0,0)	**4.57**	2.29	3.22	2.34	2.28
target	π_u^c	deterministic	1/10	(-1,1)	**4.64**	4.48	3.87	2.08	2.02
target	π_u^d	deterministic	1/3	(-5,5)	4.84	6.31	**6.41**	3.60	3.73
target	π_u^e	random	0.3	(-1,1)	3.80	4.76	4.32	**5.10**	**5.07**

Table 1: On the left part of the table, description of the source and target users. On the right, the average return $\gamma^T R^i(s_T^i)$ yielded when opposing users with the policies optimised for other users. The best policy is displayed in bold.

of the j^{th} observation, in matrix X_a. The transposition of these notations into the Transfer Learning problem gives:

$$\mathscr{L}_\Phi : (\mathcal{S} \times \mathscr{A} \times \mathcal{S} \times \mathbb{R})^N \xrightarrow{FQI} \mathbb{R}^{|\Phi|} \quad (11)$$

In the experiment, the feature vector Φ_a is a 6-dimensional vector composed of the following features for each action:

- The constant feature 1.
- The utility loss between the last proposed time-slot and the next one.
- The square of the previous feature.
- The number of time-slots which can still be proposed.
- The length of the dialogue.
- The speech recognition score.

$\mathscr{A}$ is defined according to notations in Subsection 3.1 as follows:

- ACCEPT(τ) $\Leftrightarrow$ the same as in Subsection 3.1.
- REFINSIST(τ_k) $\Leftrightarrow$ REFPROP(τ_k), with τ_k equal to the last proposed time-slot by the system.
- REFNEWPROP(τ_{k+1}) $\Leftrightarrow$ REFPROP(τ_{k+1}), with τ_{k+1} the preferred one after τ_k.
- REPEAT $\Leftrightarrow$ the same as in Subsection 3.1.

4. EXPERIMENT AND RESULTS

4.1 Experiment specifications

In the experiment, there are 5 source users and 5 target users whose characteristics are described in Table 1. The environment is fully cooperative ($\alpha = 1$) with discount factor $\gamma = 0.9$ and sentence error rate $p_{SER} = 0.3$.

At first, learning is performed individually on the sources, and the optimal policy at the end of the learning phase is saved to be used for UCB-based source selection, as well as the transitions for later transfer. With Fitted Q-Iteration, the policy is updated every 500 dialogues for a total of 5000 dialogues to ensure convergence. An ϵ-greedy policy is used with $\epsilon = \frac{1}{2j}$ where j is the iteration index. In Table 1, all the policies learnt on source users are tested on all the users – both source and target – to highlight the importance of user adaptation in our appointment scheduling game. Indeed, using a policy learnt on a source user on another user can lead to a very low return if the users are too different. In particular, using a policy learnt from a random user on a deterministic user is highly inefficient, since every REFINSIST(τ_k) act would yield a task failure. From the second part of the table, it appears that some source users are more efficient than others when their policy is applied to a target.

The features used for statistics-based source selection are the following: average speech recognition score, acceptance rate, acceptance rate after insisting, a proxy for the correct recognition rate as a function of the score. The only time when the system knows if it understood the right slot is when it accepts a request. Hence, the correct recognition rate is only updated when this situation happens. In order to have different statistics for different score values, three different statistics are collected depending on the associated speech recognition score : under 0.6, between 0.6 and 0.8, above 0.8. The correct recognition rate is biased by the policy: if the system learns to accept only proposals with recognition scores higher than 0.9, then, the "above 0.8" statistics is biased in fact towards "above 0.9" statistics.

For each new user, five algorithms are tested:

- No Transfer: without any transfer, learning is performed from scratch.
- Statistics-based (TS): with statistics-based source selection and density-based transition selection ($\eta = 0.3$).
- UCB-Based (TS): with UCB-based source selection and density-based source transition selection ($\eta = 0.3$).
- UCB-Based (noTS): with UCB-based source selection but constant source transition transfer from $n_{transfer} = 200$ dialgoues and no transition selection.
- UCB-Based (TR): with UCB-based source selection but linearly replaced source transition transfer from $n_{transfer} = 200$ dialgoues and no transition selection.

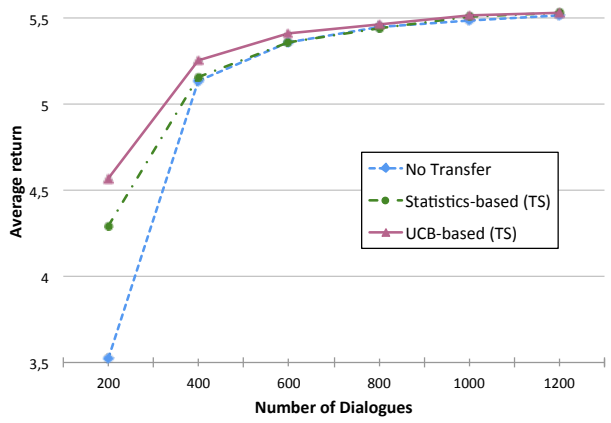

Figure 2: Learning curves comparison of algorithms for source selection.

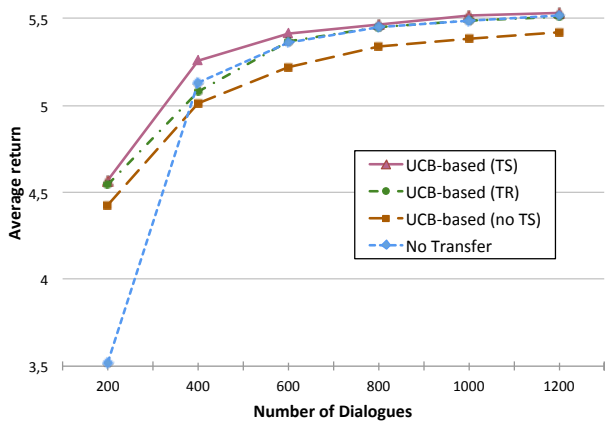

Figure 3: Learning curves comparison of algorithms for transition selection and transfer.

All those algorithms use Fitted Q-Iteration to optimise their policies. They are updated every $batchSize = 200$ dialogues with a total of $nbBatch = 6$ batches, and an ϵ-greedy policy is used with $\epsilon = \frac{1}{2^j}$. In both transfer methods, there is a first phase of source selection over $n_{first} = 50$ dialogues. The policy is then updated with Fitted Q-Iteration and used for 150 dialogues before making a second update. Then an update is made every 200 dialogues like with standard Fitted Q-Iteration.

4.2 Results and discussions

Numerical results are reported in Table 2. The values in the table represent the average returns over the first 1000 dialogues and over 100 runs for the five target users. The results show a robust significant improvement of the scores for the algorithm including both the contributions from this article: UCB source selection and transition selection. Figures 2 and 3 enable a more precise analysis of relative performances on jumpstart, learning speed and asymptote.

Figure 2 shows the difference between learning from scratch and using both kinds of source selection for target users. The first point in the figure represents the average reward of $n_{first} = 50$ random dialogues used to select the source and $200 - n_{first} = 150$ dialogues learnt with the batch from the chosen user mixed with the first dialogues of the new user. These two phases are averaged together in the graph to keep the same number of dialogues per point. Both transfer algorithms perform dramatically better than without transfer at the beginning of the learning procedure. Indeed, the policies used during the source selection phase are far more efficient than a random policy, which is used when no transfer is being used; and, transferring transitions allows to make the first update earlier. UCB was considered first because it was parameter free, and by its ability to improve immediate jumpstart, but the gain from using UCB for source selection over the statistics-based method remains clear even after two or three iterations, indicating that it is also better than the statistics-based method at selecting the source.

Figure 3 compares the three variants of the transition transfer. It appears that using transition selection, and keeping dialogues from the source in the batch are the best options. Indeed, adding an constant number of dialogues in the batch impairs both the learning speed since the start and the asymptotic performance until the end, as unwanted information is added and kept, which causes a bigger inertia from the source. Not keeping source dialogues in the batch has no effect on asymptotic performance but it impairs the learning speed (as defined in Figure 1) as the system has to re-explore the transitions that were dismissed.

Algorithm	(a)	(b)	(c)	(d)	(e)	avg
No Transfer	5.35	4.47	4.65	5.75	4.68	4.98
Statistics-based (TS)	5.55	4.58	5.00	**5.99**	4.61	5.15
UCB-based (TS)	**5.64**	**4.63**	**5.16**	**5.99**	**4.78**	**5.24**
UCB-based (noTS)	5.36	4.44	4.86	**6.00**	4.71	5.07
UCB-based (TR)	5.56	4.59	5.04	5.95	**4.78**	5.18

Table 2: Average returns for the first 1000 dialogues over 100 trials. For each target user, the significantly best results are displayed in bold.

5. CONCLUSION AND FUTURE WORK

This article coped with user adaptation in Spoken Dialogue Systems. The system is cast as a Markov Decision Process and Transfer Learning is applied in order to boost reinforcement learning returns, both in terms of jumpstart and learning speed. Empirical results on a simulated negotiation dialogue game show strong and significant performance gains on both sides.

Two novel contributions were presented in this article. Firstly, a UCB-based method to select a source showed three main advantages in comparison with the state of the art: it is model-free, it has a better performance during jumpstart, and it improves speed convergence thanks to a better source selection. Secondly, the source transitions were selected in order to cover the whole space of state-action couples. This method proved to learn fast without compromising the asymptotic convergence.

As future work, we plan to gather dialogues with real users on this game and analyse the user strategy profiles. We are also interested in studying the co-learning impact on this game and adapt the framework to non stationary behaviours (probably with Stochastic Games [33, 2]).

REFERENCES

[1] P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47(2):235–256, May 2002.

[2] M. Barlier, J. Perolat, R. Laroche, and O. Pietquin. Human-machine dialogue as a stochastic game. In *Proceedings of the 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue (Sigdial)*, 2015.

[3] W. Böhmer, J. T. Springenberg, J. Boedecker, M. Riedmiller, and K. Obermayer. Autonomous learning of state representations for control. *KI-Künstliche Intelligenz*, pages 1–10, 2015.

[4] J. L. Carroll and K. Seppi. Task similarity measures for transfer in reinforcement learning task libraries. In *Neural Networks, 2005. IJCNN'05. Proceedings. 2005 IEEE International Joint Conference on*, volume 2, pages 803–808. IEEE, 2005.

[5] I. Casanueva, T. Hain, H. Christensen, R. Marxer, and P. Green. Knowledge transfer between speakers for personalised dialogue management. In *Proceedings of the 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue (Sigdial)*, 2015. preprint.

[6] S. Chandramohan, M. Geist, and O. Pietquin. Optimizing spoken dialogue management with fitted value iteration. In *Proceedings of the 10th Annual Conference of the International Speech Communication Association (Interspeech)*, pages 86–89, 2010.

[7] C. Do and A. Y. Ng. Transfer learning for text classification. In *Proceedings of the 19th Annual Conference on Neural Information Processing Systems (NIPS)*, 2005.

[8] I. Efstathiou and O. Lemon. Learning non-cooperative dialogue behaviours. In *15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SigDial)*, page 60, 2014.

[9] L. El Asri, R. Laroche, and O. Pietquin. The negotiation dialogue game. In *Proceedings of the Seventh International Workshop on Spoken Dialogue System (IWSDS 2016)*, 2016.

[10] L. El Asri, R. Lemonnier, R. Laroche, O. Pietquin, and H. Khouzaimi. Nastia: Negotiating appointment setting interface. In *Proceedings of the 9th Edition of Language Resources and Evaluation Conference (LREC)*, 2014.

[11] Y. Engel, S. Mannor, and R. Meir. Reinforcement learning with gaussian processes. In *Proceedings of the 22nd International Conference on Machine Learning (ICML)*, pages 201–208. ACM, 2005.

[12] M. S. English and P. A. Heeman. Learning mixed initiative dialogue strategies by using reinforcement learning on both conversants. In *Proceedings of the conference on Human Language Technology (HLT)*, 2005.

[13] E. Even-Dar, S. Mannor, and Y. Mansour. Action elimination and stopping conditions for reinforcement learning. In *Proceedings of the 20th International Conference on Machine Learning (ICML)*, pages 162–169, 2003.

[14] F. Fernández and M. Veloso. Probabilistic policy reuse in a reinforcement learning agent. In *Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems*, pages 720–727. ACM, 2006.

[15] E. Ferrante, A. Lazaric, and M. Restelli. Transfer of task representation in reinforcement learning using policy-based proto-value functions. In *Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems-Volume 3*, pages 1329–1332. International Foundation for Autonomous Agents and Multiagent Systems, 2008.

[16] M. Gašić, C. Breslin, M. Henderson, D. Kim, M. Szummer, B. Thomson, P. Tsiakoulis, and S. Young. Pomdp-based dialogue manager adaptation to extended domains. In *Proceedings of the 14th Annual Meeting of the Special Interest Group on Discourse and Dialogue (Sigdial)*, 2013.

[17] M. Gašić, D. Kim, P. Tsiakoulis, C. Breslin, M. Henderson, M. Szummer, B. Thomson, and S. Young. Incremental on-line adaptation of pomdp-based dialogue managers to extended domains. In *Proceedings of the 14th Annual Conference of the International Speech Communication Association (Interspeech)*, 2014.

[18] K. Georgila, C. Nelson, and D. Traum. Single-agent vs. multi-agent techniques for concurrent reinforcement learning of negotiation dialogue policies. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 500–510, Baltimore, Maryland, June 2014. Association for Computational Linguistics.

[19] K. Georgila and D. R. Traum. Reinforcement learning of argumentation dialogue policies in negotiation. In *Proceedings of the 11th Annual Conference of the International Speech Communication Association (Interspeech)*, pages 2073–2076, 2011.

[20] B. Hengst. *Discovering hierarchy in reinforcement learning*. University of New South Wales, 2003.

[21] H. Khouzaimi, R. Laroche, and F. Lefevre. Optimising turn-taking strategies with reinforcement learning. In *Proceedings of the 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue (Sigdial)*, 2015.

[22] P. Langley. Transfer of knowledge in cognitive systems. Paper presented at the workshop on Structural Knowledge Transfer for Machine Learning at the Twenty-Third International Conference on Machine Learning (ICML), 2006.

[23] R. Laroche and A. Genevay. The negotiation dialogue game. In *Proceedings of the Seventh International Workshop on Spoken Dialogue System (IWSDS 2016)*, 2016.

[24] R. Laroche, G. Putois, P. Bretier, M. Aranguren, J. Velkovska, H. Hastie, S. Keizer, K. Yu, F. Jurcicek, O. Lemon, and S. Young. D6.4: Final evaluation of classic towninfo and appointment scheduling systems. *Report D6*, 4, 2011.

[25] A. Lazaric. Transfer in reinforcement learning: a framework and a survey. In *Reinforcement Learning*, pages 143–173. Springer, 2012.

[26] A. Lazaric, M. Restelli, and A. Bonarini. Transfer of samples in batch reinforcement learning. In *Proceedings of the 25th International Conference on Machine Learning (ICML)*, pages 544–551. ACM, 2008.

[27] O. Lemon and O. Pietquin, editors. *Data-Driven Methods for Adaptive Spoken Dialogue Systems: Computational Learning for Conversational Interfaces* . Springer, November 2012.

[28] E. Levin and R. Pieraccini. A stochastic model of computer-human interaction for learning dialogue strategies. In *Proceedings of the 5th European Conference on Speech Communication and Technology (Eurospeech)*, 1997.

[29] S. Mahadevan and M. Maggioni. Proto-value functions: A laplacian framework for learning representation and control in markov decision processes. *Journal of Machine Learning Research*, 8(2169-2231):16, 2007.

[30] N. Mehta, S. Natarajan, P. Tadepalli, and A. Fern. Transfer in variable-reward hierarchical reinforcement learning. *Machine Learning*, 73(3):289–312, 2008.

[31] S. J. Pan and Q. Yang. A survey on transfer learning. *Knowledge and Data Engineering, IEEE Transactions on*, 22(10):1345–1359, 2010.

[32] F. Sadri, F. Toni, and P. Torroni. Dialogues for negotiation: agent varieties and dialogue sequences. In *ATAL*, pages 405–421. Springer, 2001.

[33] L. S. Shapley. Stochastic games. *Proceedings of the National Academy of Sciences of the United States of America*, 39(10):1095, 1953.

[34] S. P. Singh, M. J. Kearns, D. J. Litman, and M. A. Walker. Reinforcement learning for spoken dialogue systems. In *Proceedings of the 13th Annual Conference on Neural Information Processing Systems (NIPS)*, 1999.

[35] R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

[36] R. S. Sutton, D. Precup, and S. Singh. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial intelligence*, 112(1):181–211, 1999.

[37] M. E. Taylor and P. Stone. Representation transfer for reinforcement learning. In *AAAI 2007 Fall Symposium on Computational Approaches to Representation Change during Learning and Development*, 2007.

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

[39] S. Young, M. Gašić, B. Thomson, and J. D. Williams. POMDP-based statistical spoken dialog systems: A review. *Proceedings of the IEEE*, 101(5):1160–1179, 2013.