2018 IEEE-RAS 18th International Conference on Humanoid Robots (Humanoids)
Beijing, China, November 6-9, 2018

Skill transfer for mediated interaction learning

Sascha Fleer and Helge Ritter

Abstract— **We investigate the effectiveness of transfer learning for accelerating shallow and deep reinforcement learning of "mediated interaction tasks". In these tasks, the desired effects cannot be created through direct interaction, but instead require the learner to discover how to exert suitable effects on the target object through involving a suitable "mediator object". We focus on the case where transfer learning is applied to generalize experiences from source tasks that are solvable through direct, unmediated interaction, to target tasks that require mediated interaction for their solution. We find that transfer learning employed in this context leads to a significant acceleration of learning. A refinement of the basic transfer learning strategy that is motivated from the principle of scaffolding in psychology leads to further improvements, totalling to an overall speed-up factor of almost one order of magnitude for a reinforcement learner solving an "extension-of-reach" task in a 2D world with simulated physics.**

I. INTRODUCTION

Many daily actions, such as pushing an object with a stick, pouring the contents of a mug, or accessing a book in a drawer, require some form of "mediated interaction" where the agent can interact with the target object only after actively preparing access via the intermediate use of some auxiliary "mediator object". Closely related is tool use, a learned capability found in humans, but also in some other species, such as some primates or birds [1]. It has been linked to higher cognition and, obviously, is also a desirable capability for robots to become more adept at many daily tasks typically arising in human environments [2][3].

The interaction of humanoid robots and humans is a challenging task which not only requires the fast learning of new assigned duties but also the ability to learn in ways that enable the human to easily instruct the robot. If the robot's morphology is also resembling a human, tool affordances, e.g. their use as a mediator object within a tasks context, is also an important aspect that has to be taken into account [2][4].

Reinforcement learning has been considered as an attractive method to enable robots to master mediated interaction tasks. While conceptually very attractive, a major problem arises from often prohibitively large numbers of required learning steps to achieve reasonable performance. Especially in cases where deep neural networks are involved, the long training times might render the learning impossible when real robots are involved within the learning process.

Transfer learning has been suggested as a partial remedy [5]. It has been shown to be very effective to initialize a reinforcement learner from the results of previous learning of simpler *source tasks* which share structure with or are suited as building blocks for the original *target task* of the reinforcement learner. Examples include maze navigation tasks [6], simulated robot soccer [7] and also first attempts to learn tasks using a physical robot [8]. While most transfer learning schemes are not paying much attention to the fashion in which the source tasks are provided, it can be combined with *curriculum learning* [9]. This method is claiming that a thoughtfully chosen sequence of source tasks can further enhance the learning process [10] and has become an attractive approach to tackle complex problems within the field of deep reinforcement learning [11][12].

In the present contribution, we investigate the effectiveness of transfer learning when the source tasks are instances that are solvable through direct, *unmediated interaction,* while the target tasks are instances that can only be solved through *mediated interaction.* Thus, our focus is how transfer learning can help to facilitate mediated interaction learning from experiences from simpler tasks that are similar, but permit direct, unmediated solutions.

As a further addition, we report results that the effectiveness of transfer learning can be further enhanced by applying curriculum learning. The invented method that is used to create the task sequences is loosely motivated from a concept known in psychology under the name of *scaffolding* [13]: it denotes a particular strategy of choosing source tasks such that the level of difficulty of each new source task increases with the proficiency of the learner so that it always stays within what psychologists denote as the learner's *"optimal zone of proximal development"*.

For our study we use a simulated 2D interaction scenario in which an artificial agent is able to interact with different objects. We find that within this simplified, yet physically realistic toy world, the integration of transfer learning is able to speed up the learning process of a "extension-of-reach task" by about an order of magnitude. Non-linear function approximators like neural networks are specifically designed to represent very non-linear problems, and it might be hard to justify the increase in parameters and training complexity at a first glance. As linear function approximators are, however, very sensitive on changes of the parameters and also on the learning task, it is advantageous to use non-linear function approximators for an enhanced version of the studied interaction scenario (e.g. varying the used tools or using a real/simulated robot for the control of the objects). Thus, showing the efficiency of the proposed strategies when combined with non-linear Q-Learning in this "simpler" scenario can be a good indicator that the proposed approaches

The authors are with the Neuroinformatics Group, EXC Cognitive Interaction Technology (CITEC), Bielefeld University, Germany. sfleer, helge@techfak.uni-bielefeld.de

978-1-5386-7282-2/18/$31.00 ©2018 IEEE

are also increasing the learning efficiency in more complex ones, where utilizing non-linear function approximators is a reasonable choice.

Section II gives a short introduction of Q-learning with linear and non-linear function approximation which is used in this work. The next section III introduces the learning domain and some additional information on the composure of the environment. Section IV then explains the transfer learning scheme in detail, together with a way of additional improvement and stability through the scaffolding based approach. After describing the learning scenarios, the experiments and the used learning algorithms in section V, we present an evaluation of the agent's learning performance for the proposed approaches (section VI). Section VII concludes with a final discussion of the achieved results, together with some suggestions to future work.

II. REINFORCEMENT LEARNING

Reinforcement learning is a well-known class of machine learning algorithms for solving sequential decision making problems through maximization of a cumulative scalar reward signal [14].

We use the standard formulation of a Markov decision process defined by tuple $(S, A, P^A, R, \gamma, S_0)$, where S denotes the set of states and A the set of admissible actions. P^A is the set of transition matrices, one for each action $a \in A$ with matrix elements $P^a_{s,s'}$ specifying the probability to end up in state s' after taking action a when in state s. Finally, $R : S \times A \to \mathbb{R}$ is a scalar valued reward function, γ the discount factor and $S_0 \subseteq S$ is the set of starting states.

Reinforcement algorithms are now aiming for the policy $\pi : S \to A$ that maximizes the discounted future reward. To find this optimal policy, the state-action value

$$Q^\pi(s_t, a_t) = \mathbb{E}^\pi \left[R(s_t, a_t) + \sum_{k=t+1}^{\infty} \gamma^{k-t} R(s_k, \pi(s_k)) \right]$$

is defined. $Q^\pi(s_t, a_t)$ has the interpretation of the discounted future reward, expected from *following the current policy* π *after* taking a single freely choosable (and possibly sub-optimal) action a_t from state s_t. The discount factor $\gamma \in [0, 1)$ balances the weighting between present rewards and rewards that lie increasingly in the future.

Thus, the agent should follow the policy $\pi(s_t) = \max_a Q^\pi(s_t, a)$ that chooses the action at each time step which maximizes the current state-action value.

If the dimension of the state space is huge, which is the case in most complicated learning scenarios, a common approach is to approximate $Q(s, a)$ by using a linear function

$$Q(s, a, \vec{w}) = \vec{w}^\intercal \cdot \vec{\Phi}(s, a)$$

with a weight vector $\vec{w} \in \mathbb{R}^n$ and a function $\vec{\Phi}(s, a) \in \mathbb{R}^n$ of lower dimensional features that represents the state-action pair (s, a).

The optimal state-action value can then be approximated by iteratively minimizing the squared Bellman error

$$\mathscr{L}_t(\vec{w}_t) = \mathbb{E}\left[\left(r_t + \gamma \cdot \max_a Q(s_{t+1}, a; \vec{w}_t) - Q(s_t, a_t; \vec{w}_t) \right)^2 \right].$$

This leads to a stochastic gradient descent update for the weights $\vec{w}$ at time step $t+1$, defined as

$$\vec{w}_{t+1} = \vec{w}_t + \alpha_t \cdot \left[r_t + \gamma \cdot \max_a Q(s_{t+1}, a; \vec{w}_t) - Q(s_t, a_t; \vec{w}_t) \right] \cdot \nabla_{\vec{w}_t} Q(s_t, a_t; \vec{w}_t). \quad (1)$$

With the help of (1), $Q(s, a)$ can be updated iteratively, while the update size is determined by a (typically decreasing) learning rate $\alpha_t \in [0, 1)$.

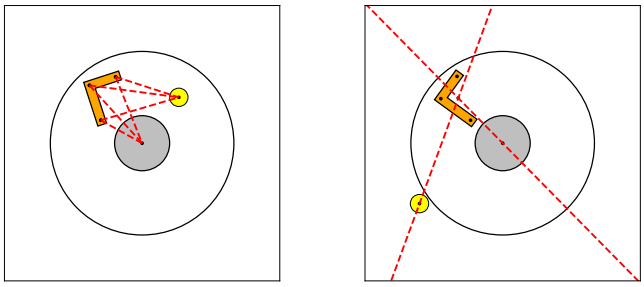
One drawback of this kind of linear approximation is that the type of function $\vec{\Phi}(s, a)$ representing the state-action pairs has to be chosen very carefully to achieve a good learning performance. The difficulty of finding a reasonable representation increases with the complexity of the given state space.

One way out of this dilemma is to replace the linear function that represents the given state with a non-linear function like a neural network. Neural networks are able to easily abstract the most complex functions by a hierarchical composition of low-level abstractions. In a deep neural network multiple layers of neurons are stacked together, where each layer computes abstract representations of the input data with increasing complexity. The combination of deep neural networks with reinforcement learning led to many breakthroughs as learning to play video games [15] or mastering the challenging board game GO [16]. One popular deep reinforcement algorithm is *deep Q-learning* which is presented into detail in [15]. This algorithm approximates the state-action value function $Q(s, a)$ with a deep neural network. In order to train the deep Q-learner, the experienced transitions (s_t, a_t, r_t, s_{t+1}) are stored in a replay buffer which is then used to train the network weights according to (1) using mini-batch gradient descent.

III. LEARNING DOMAIN

As our testbed for investigating skill transfer in mediated interaction learning we employ a 2D world with physics simulated with the open source `Box2D` physics engine [17]. The world is illustrated in Figure 1a and consists of an agent, a disc-shaped "target object" (yellow) and an L-shaped object "mediator object" ("tool", orange). The task of the agent is to bring the goal object into the shaded circle in the center ("goal area"). To this end, the agent can at each time step "pick" the goal object or the mediator object and exert a (discretized) force/torque at the chosen picking location. This allows the agent to move the chosen object about a fixed distance of one unit per time step within the admissible interaction range which has a radius of 10 units. The four directions of the movement are determined by the non-static coordinate system in Figure 1b, where the objects can be navigated along the coordinate axis. Additionally the agent is able to rotate the tool around the active picking location by $\pi/4$ per time-step. Instead of using friction, a linear and angular damping factor is implemented.

Picking locations are discretized and fixed at the objects (see Figure 1a): the goal object offers a single picking location at its center, the mediator object offers three picking



(a) Construction of the sensory input. The five dots indicate the five different picking locations. It is also an example of a "single-object interaction task".

(b) Illustration of the used sensorimotor coordinate system to manipulate the objects position within the simulation world. It is also an example of a "mediated interaction task".

Fig. 1: Illustration of the presented simulation world. The picking locations are indicated by black dots.

locations, two at its ends and one in the middle. Furthermore, there is an additional picking location in the center of the domain which deals as an unbiased starting location for the agent and is further integrated to be an absorbing state that increases the stability of the applied learning algorithms.

We assume that the agent has a simple relational perception of the world state consisting of the six scalar distances between the three picking locations $\vec{M}_i$ on the mediator-object and the center of the target-object $\vec{T}$, i.e. $|\overrightarrow{M_i T}|$, and the three distances of the picking locations $\vec{M}_i$ and the domain's origin $\vec{O}$. Additionally, the sensory representation includes a 5-dimensional binary vector $\vec{A}_{pos}$ that encodes the information of the agent's current picking choice.

These distances, visualized by the dotted red lines in Figure 1a, along with the binary picking information, give rise to the state vector

$$\vec{s} = \left(|\overrightarrow{M_1 T}|, |\overrightarrow{M_2 T}|, |\overrightarrow{M_3 T}|, |\overrightarrow{M_1 O}|, |\overrightarrow{M_2 O}|, |\overrightarrow{M_3 O}|, \vec{A}_{pos} \right)^\top . \tag{2}$$

However, the agent can only reach picking locations that lie inside the circular area. Therefore, when the goal object is outside the circle, the agent must first "discover" that the mediator object can be used to extend the agent's reach beyond the circle boundary.

Consequently, the agent can choose between 11 actions to interact with the environment. Learning occurs in discrete episodes, each episode being limited to 100 interaction-steps. If the agent is able to navigate the target object in the goal area[1], it receives a fixed reward of $r = 10$. By this kind of task definition the influence on the agent's learning process is minimized, but has the disadvantage that the agent takes very long to start learning as it has at least once to solve the problem by chance to gain its first reward. Additionally, artificial noise is integrated into the system that makes the agent execute a random action with a probability of 0.1. The additional noise term adds a stochastic component to the deterministic learning domain that disturbs the agent occasionally with a sub-optimal action.

[1] i.e. the distance to the domains origin is smaller than the radius of the goal area

IV. SKILL TRANSFER WITH SCAFFOLDING FOR ACCELERATING LEARNING OF MEDIATED CONTROL

We present two strategies for accelerating learning of mediated control. *Strategy 1*, formalized in Algorithm 1, uses only skill transfer based on prior learning of a number of randomly selected, simpler source tasks before switching to learning instances of the target task. *Strategy 2* complements *Strategy 1* with an approach that replaces the random selection of the source tasks with a structured selection that restricts the choice of new source tasks to a vicinity of the set of already solved source tasks. This vicinity is defined with the help of a suitable similarity metric in the source domain. It is formalized in Algorithm 2.

Both strategies assume a simple and generic structure of the learning domain: learning should occur in a sequence of episodes, each episode focused on a single task instance, attempting to solve it in a number of consecutive time-steps (number of actions) and terminating after success or when a step limit is exceeded. After termination, another episode starts. Finally, we assume that source and target tasks are from the same domain, so that both can be handled by the same learning process.

Under these assumptions, we can represent each task (irrespective of being a source or target task) by a pair $\vec{c} = (\vec{s}, \vec{t})$, where $\vec{s}$ represents the task in the sensor space of the agent, and $\vec{t}$ represents the task in some task space coordinates (which usually differ from sensor coordinates). We denote by C the set of solvable source tasks that is provided for the source learning stage and by C_{solved} the subset of source tasks that already have been solved at a particular time step. The number of to-be-solved source tasks, i.e. the number of elements in C is given by N_{source}.

Both strategies are motivated from the typical characteristic of mediated control: reaching the goal requires to suitably concatenate a number of subskills. Each subskill refers to the goal object or to one of the (single or more) mediator objects. To solve such kinds of task, the agent has to learn these subskills and bring them together in a suitable sequence. For instance, in the learning domain considered in this paper, a subskill is to learn moving the goal object directly into the goal region when it is sufficiently close to it. For complex tasks, each subskill may itself require a decomposition, requiring the agent to organize an entire hierarchy of subskills. This existence of subskills can be exploited by pre-learning the most essential low-level ones, e.g., the identification and manipulation of the target object as well as the recognition of the goal conditions. Then, instead of learning source tasks up to perfection, the agent only learns them a small number of times and uses the imperfectly learned subskills for a "lightweight initialization" of the learning process of the target task. This also prevents specialization on the source tasks to interfere with the learning of the target task. In the target task the agent should learn, based on the knowledge

Algorithm 1: Strategy 1 - Transfer learning for mediated interaction tasks

```
Data: Set C of N_source clustered source tasks
for learning episode = 1, M do
    if C ≠ ∅ then
        Random sample task instance from C
        start learning episode
        if Agent solves task then
            remove task from C
    else
        start episode and learn the target task
```

Algorithm 2: Strategy 2 - Scaffolded transfer learning for mediated interaction tasks

```
Data: Set C of N_source clustered source tasks
for learning episode = 1, M do
    if C ≠ ∅ then
        if C_solved = ∅ then
            Sample task c⃗ from C
        else
            Sample random task k⃗ ∈ C_solved
            Among unsolved source tasks C:
            find task c⃗ ∈ C that is similar to k⃗
    else
        Sample random task c⃗ for the target task
    start episode for learning c⃗
    if Agent solves task c⃗ and C ≠ ∅ then
        add c⃗ to C_solved
        remove c⃗ from C
```

it has achieved in the source task, more intermediate skills like how to use objects as tools or object-object interaction.

The simple transfer learning *Strategy 1* implements this idea, switching from source task to target task learning after a given set of to-be-learned source task instances has been solved without stopping the learning process. It has a set C of N_{source} tasks as its only tunable input parameter.

Strategy 2 refines *Strategy 1* through a more structured selection strategy for the source tasks. The adopted strategy follows a fundamental principle of human learning that has become known under the name of scaffolding [13]: learning progress is accelerated when new learning instances are not selected at random, but instead are matched to the learners current proficiency. This requires a new example to be neither too easy nor too difficult for the learner. This can be achieved through a "similarity-similarity based heuristic": choosing each new learning example to be similar to one of the instances that the learner has solved previously. The simplest implementation of such a heuristic employs a suitable *similarity metric* in the domain of the task instances (cf. below). Besides similarity, further constraints (such as recency) could refine the selection further. *Strategy 2* implements the simplest variant, based on similarity with no further constraints. Therefore the next to-be-solved source task is selected by minimizing the distance to a solved source task, according to the chosen similarity metric. Similar to *Strategy 1*, its main input parameter is again a set C of to-be-solved source tasks before switching to learning the target task occurs.

An important ingredient in both algorithms is the sampling of their instances, with *Strategy 2* in addition requiring a similarity metric.

A. SENSOR AND TASK SPACE SAMPLING

Tasks are defined within the task space. Yet, the task space differs in most cases from the sensor space which is used as an input for the learning algorithm. In the worst case, the mapping from the task space to the sensor space is also not invertible. This decorrelation of task and sensor space can have a huge impact on the learning process as small deviations within the task space can have completely different effects within the sensor space. Consequently, the agent's modalities of perception, encoded in the sensor space, have to be taken under consideration during the structuring of the learning process.

Minimizing the number of required source tasks N_{source} while maintaining a uniform distribution over the relevant part of the sensor space, i.e. it is accessible through a mapping from the task space, can be realized through clustering. Therefore a large number of random tasks $\vec{c} = (\vec{s}, \vec{t})$ is generated within the task space. The next step is to cluster the tasks $\vec{c}$ with respect to $\vec{s}$. Now, for each cluster the task $\vec{c}$ with the most similar representation $\vec{s}$ in the sensor space is chosen to be in the task set C .

B. SIMILARITY METRIC

A good measure of similarity is a key ingredient for scaffolding strategies. To efficiently structure the learning process the pending tasks have to be conveniently compared to find the one which aligns best with the agents current skills. One reasonable choice is to measure the correlation between different tasks $\vec{s}$ in the sensor space by utilizing the cosine similarity

$$\text{CosSim}(u,v) = \frac{(\vec{u}\cdot\vec{v})}{||\vec{u}||_2||\vec{v}||_2}. \tag{3}$$

The best choice how to determine the correlation of the tasks is heavily depending on the structure of their embedded space and the processing afterwards. Thus the similarity of tasks is not only influenced by their own composition but also through the characteristics of the used representation and learning algorithm.

C. PERFORMANCE METRICS

If an artificial agent has to learn the interaction with objects in the real world there are often many undesired events that may happen because of suboptimal behaviour.

The objects might break or the agent itself might be damaged. So it is crucial to reach a *good performance as soon as possible* in order to minimize the chance of these events to happen.

Based on this statements two evaluation metrics are chosen to evaluate the efficiency of the designed transfer learning scheme against regular learning.

a) Total Reward Loss: The total reward loss $\mathscr{L}_R$ is defined as the sum of the differences between the optimal achievable reward $R_{\max} = 10$ and the achieved average reward $\langle R_i \rangle$ evaluated at learning step i. With the total reward loss defined as

$$\mathscr{L}_R = \sum_{i}^{T} \left[R_{\max} - \langle R_i \rangle \right],$$

the learning process with the smallest $\mathscr{L}_R$ is the one where the agent has gained the most reward and has therefore solved the tasks with the highest efficiency.

b) Time to Threshold: This metric, which is introduced in [5], measures the number of learning steps which are needed to reach a certain performance threshold. The choice of a reasonable threshold performance is depending on the studied learning scenario. As the goal in object interaction tasks is to let the agent solve them with a good performance as fast as possible to minimize errors and therefore the probability of damaging the environment, the threshold performance for this work is chosen to be at an average success rate of 80%. Although this threshold is significantly below optimal success, it is on the one hand high enough to tag the corresponding policies as successful and on the other hand low enough that the performance level can be achieved in all experiments within a reasonable amount of learning steps (see eg. Figure 3).

V. EXPERIMENTAL SETUP

Utilizing the described simulation world, two learning tasks were designed.

a) Single-Object Interaction Scenario: At the beginning of a learning episode, the tool and the target object are uniformly distributed over the simulation world inside the agent's interaction range (see e.g. Figure 1a). The agent now has to learn how to move the target object into the goal area. After successfully solving the task instance or exceeding the limit of possible interaction-steps per episode, the task starts anew with different initial object positions that are again within the agents interaction range.

b) Mediated Interaction Scenario: This "extension-of-reach scenario" is structured like the first one with the exception that the target object is distributed *outside* the border of the agent's interaction range. Now it is only possible for the agent to solve this task by learning to exploit the hook as a tool to pull the target object inside the agent's interaction range (see e.g. Figure 1b).

The "mediated interaction scenario" can efficiently be solved, using the presented learning schemes (see section IV) by transferring skills from the "single object interaction task" (Figure 1a). It assigns the same problem to the agent, i.e. navigate the target object to the goal, but under easier conditions and is therefore a good choice to be used as the source task.

At first, the results of the regular transfer learning scheme (*Strategy 1*), presented in section IV, is evaluated. The set C of the to-be-learned source task configurations are the allowed object positions $(\vec{x}_{\text{target}}, \vec{x}_{\text{tool}})$ within position- and sensor-space[2] that can arise during the "single object interaction task". The characteristic aspect of this configurations is that the target object is always within the agents reach and can be navigated to the goal area without the use of the tool. For the clustering part, 10^4 target configurations together with the cosine similarity (3) are used to generate varying number of source task configurations N_{source}. During the learning, these source task configurations are randomly presented to the agent at the beginning of each episode as described in *Algorithm 1* until the agent has solved the problem N_{source} times. Afterwards the agent directly starts to learn the target task without any delay. This implies that learning the source tasks is *not* rolled out within a separate learning process but treated as a pre-learning routine that is carried out right before learning the target task. Thus the learning steps needed to solve the source tasks are included within the learning steps of the whole learning run.

In a further step, the transfer learning scheme is extended by "scaffolding" the source task learning as described in *Algorithm 2*. Therefore, the same experiment as presented above is conducted with the improved transfer learning algorithm.

To gain more insight into the impact of different function approximations on the learning process, we evaluate the learning performance for two kinds of linear function approximators representing the sensor vector (2) and for a deep Q-learner which uses a neural network as a non-linear function approximator. Although the given learning scenario is solvable using linear Q-learning, it is worth to include a non-linear function approximator. Despite the fact that it needs longer training time, it is much more stable under changes of its hyperparameters than the linear versions, where a small change in the parameter space - especially in the configuration of the function approximator - can hardly undermine the learning process. For the case involving a linear function approximator, we test our approach using an ε-greedy Q-learning algorithm with eligibility traces and linear function approximation [18]. The first more efficient real valued representation is based on Gaussian radial basis functions [19] (RBF). It represents the first three real-valued dimensions of the state vector (2), which are the distances between the target object and the tool's picking locations, via five uniformly placed radial basis functions per dimension. The other three real-valued dimensions of (2), incorporating the distances between the picking locations of the tool and the domain's origin, are expressed via three uniformly placed radial basis functions per dimension. The standard deviation

[2]In the presented scenario, the position space corresponds to the task space and the sensor space is defined through the state vector (2).

TABLE I: List of the optimized parameters used for learning

Parameter	RBF	FSR	Parameter	DNN
ε	0.03	0.075	$\mathscr{E}$	100000
α_0	0.82	0.91	$\mathscr{E}_{\text{start}}$	5000
α_B	8459.74	59.52	Batch Size	32
γ	0.97	0.98	ε-decay	500000
λ	0.65	0.19	τ	200
			α	0.00025
			γ	0.95

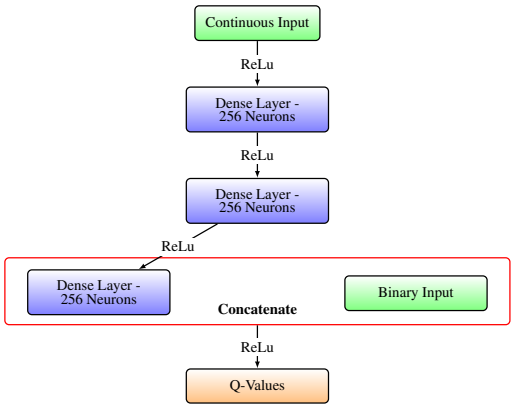


Fig. 2: Illustration of the designed deep Q-learner.

of each RBF is set to $\sigma = 1$. The binary part of the state vector is integrated using a binary representation. This kind of state representation segments the real valued state vector into 13829 quantized features. The second is a simpler binary fixed sparse representation [20] (FSR), which describes the agent current state via 310 features.

For the case involving the non-linear function approximator, we use a deep Q-learner [15]. In particular, we use a deep neural network (DNN) which is build out of 3 fully connected layers with 256 neurons and a rectified linear unit as the activation function. The used architecture is illustrated in Figure 2. Similar to the linear representation based on RBFs, we split the state vector (2) into a continuous and a binary part. The continuous part is propagated through the whole network so that suitable features for the representation of the distances between the objects and the goal are learned. The binary part only encodes the current picking location of the agent and is thus uncorrelated to the objects distances. As the binary representation can already be seen as a very condensed and efficient feature vector, it is directly concatenated with the generated features of the last layer.

The learning parameters are individually optimized for each algorithm using random search [21] followed by additional manual tuning, so that the reward within the regular learning runs without transfer learning is maximized. The optimized parameters for the linear Q-learner are the greediness ε, the eligibility factor λ, the discount factor γ, the initial learning rate α_0 and the decay parameter for the decay of the learning rate α_B (Boyan-Decay [22]) are listed in Table I. The optimized parameters for the deep Q-learner are also listed in Table I. Here, we use a constant learning rate α. The greediness is decaying linearly from 1 to 0.1. As proposed in [15], we use a target network to compute the gradient which receives a copy of the original networks parameters every τ steps. The number of transition tuples stored in the replay buffer for experience replay is given by $\mathscr{E}$, while $\mathscr{E}_{\text{start}}$ defines the minimal number of tuples that are required to start the learning process. During training, the weights are updated using the Adam algorithm [23].

For evaluating the efficiency of the learning processes, we depict the average reward per episode $\langle R \rangle$ that is received by the agent as a function of the number of learning steps. To compute $\langle R \rangle$, the learning performance under the current policy was evaluated over 100 episodes for each of the 25 evaluated data points. Note that during these performance tests, the exploration parameter ε is set to 0 and only the domain-specific noise is present. The results were then averaged over 20 distinct learning runs, where the standard deviation of the mean is used as the error. By using the two evaluation metrics, presented in section IV, the learning process of the mediated interaction scenario without transfer learning can now be compared with the transfer learning schemes into more detail.

VI. RESULTS

The results for a different number of source task configurations N_{source} are listed in Table II for the real valued representation, in Table III for the binary representation and in Table IV for the deep neural network. The first notable point listed in both tables of the linear function approximators is that the agent has just to solve the source task one single time ($N_{source} = 1$) to get a favorable seeding of the Q-values that is sufficiently good enough to speed up the target task. Increasing the number of source task problems further optimizes the learning by reducing the reward loss $\mathscr{L}_R$. The best performance, with a minimal $\mathscr{L}_R$ can then be achieved for $N_{source} = 50$ (RBF), $N_{source} = 15$ (FSR) and $N_{source} = 100$ (DNN). If N_{source} gets large, the learning process becomes too specialized on the source task. This phenomenon leads to an increase of $\mathscr{L}_R$ and the learning time to reach the performance threshold up to a point where the transfer learning process undermines the learning of the target task. The extreme case of $N_{\text{source}} = 5000$ suppresses the learning of the target task for the learning process with the binary representation and the deep neural network.

The learning performance of the deep Q-learner is shown in Figure 3. As the deep Q-learner has much more weights to train than the linear Q-learner, the learning is slower than for the other two cases. Nevertheless, our (regular) transfer learning scheme (*Strategy 1*) enables the agent to reach the threshold performance after $\approx 220 \cdot 10^3$ learning steps and a final performance of $\langle R \rangle \approx 9$ within the limit of $500 \cdot 10^3$ steps. The regular learning only reaches a final performance of $\langle R \rangle \approx 5$ which corresponds to a probability of only 50% to successfully solve the task.

For the linear Q-learning, adding scaffolding to the transfer learning scheme (*Strategy 2*) is able to reduce the time

TABLE II: Evaluation of the learning process over $500 \cdot 10^3$ learning steps, using the real valued state representation (RBF).

Used Scheme	N_{source}	$\mathscr{L}_R$	Time to Threshold
without transfer	0	105.1 ± 15.08	$\approx 400 \cdot 10^3$ steps
regular transfer	1	44.79 ± 8.16	$\approx 110 \cdot 10^3$ steps
	15	31.18 ± 5.62	$\approx 80 \cdot 10^3$ steps
	50	$\mathbf{28.985 \pm 5.33}$	$\mathbf{\approx 70 \cdot 10^3}$ **steps**
	100	33.035 ± 5.67	$\approx 80 \cdot 10^3$
	5000	60.1 ± 6.11	$\approx 140 \cdot 10^3$ steps
scaffolded transfer	1	42.32 ± 5.67	$\approx 110 \cdot 10^3$ steps
	15	33.71 ± 4.33	$\approx 80 \cdot 10^3$ steps
	50	30.84 ± 4.38	$\approx 60 \cdot 10^3$ steps
	100	$\mathbf{29.97 \pm 4.02}$	$\mathbf{\approx 60 \cdot 10^3}$ **steps**
	5000	63.04 ± 4.99	$\approx 130 \cdot 10^3$ steps

TABLE III: Evaluation of the learning process over $500 \cdot 10^3$ learning steps, using the binary state representation (FSR).

Used Scheme	N_{source}	$\mathscr{L}_R$	Time to Threshold
without transfer	0	92.24 ± 10.13	$\approx 200 \cdot 10^3$ steps
regular transfer	1	64.13 ± 9.37	$\approx 140 \cdot 10^3$ steps
	15	50.45 ± 5.20	$\approx 60 \cdot 10^3$ **steps**
	50	$\mathbf{47.63 \pm 4.68}$	$\mathbf{\approx 60 \cdot 10^3}$ **steps**
	100	54.85 ± 5.86	$\approx 120 \cdot 10^3$
	5000	246.13 ± 4.71	–
scaffolded transfer	1	64.13 ± 9.37	$\approx 130 \cdot 10^3$ steps
	15	$\mathbf{45 \pm 4.05}$	$\mathbf{\approx 40 \cdot 10^3}$ **steps**
	50	53.435 ± 4.82	$\approx 100 \cdot 10^3$ steps
	100	58.96 ± 4.64	$\approx 90 \cdot 10^3$ steps
	5000	242.45 ± 6.98	–

needed to reach the threshold performance again. The evaluated results for regular learning and scaffolded transfer learning for the RBF representation are shown in Figure 4. Both learning processes are able to achieve an average reward of $\langle R \rangle \approx 10$. This corresponds to a probability of nearly 100% to successfully solve the mediated interaction task. The learning curve in Figure 4 shows that the time to reach a stable average success rate of 80% in the complex "mediated interaction scenario" can be reduced from $\approx 400 \cdot 10^3$ to $\approx 60 \cdot 10^3$ learning steps by scaffolded transfer learning, while regular transfer learning reduces the learning steps to $\approx 70 \cdot 10^3$.

For the binary representation (FSR), scaffolded transfer learning is also able to again reduce the required training time from $\approx 60 \cdot 10^3$ to $\approx 40 \cdot 10^3$ steps. The learning performance of scaffolded transfer learning is compared to regular learning in Figure 5.

For the deep Q-learner, the required time to reach the threshold performance is increasing from $\approx 220 \cdot 10^3$ to $\approx 310 \cdot 10^3$ steps when using the scaffolded transfer learning approach. However, the scaffolded approach is much more robust on changes of N_{source}, as now the learning runs with $N_{\text{source}} = 50$ are also able to reach a success rate above 80%.

TABLE IV: Evaluation of the learning process over $500 \cdot 10^3$ learning steps, using the deep neural network (DNN).

Used Scheme	N_{source}	$\mathscr{L}_R$	Time to Threshold
without transfer	0	201.41 ± 16.37	–
regular transfer	1	158.13 ± 17.90	–
	15	123.38 ± 15.00	$\approx 300 \cdot 10^3$ steps
	50	149.02 ± 17.80	–
	100	$\mathbf{106.18 \pm 10.63}$	$\mathbf{\approx 220 \cdot 10^3}$ **steps**
	5000	233.36 ± 10.62	–
scaffolded transfer	1	201.78 ± 16.60	–
	15	$\mathbf{126.26 \pm 13.55}$	$\mathbf{\approx 310 \cdot 10^3}$ **steps**
	50	139.76 ± 15.07	$\approx 360 \cdot 10^3$ steps
	100	134.265 ± 15.22	$\approx 380 \cdot 10^3$ steps
	5000	223.3 ± 12.07	–

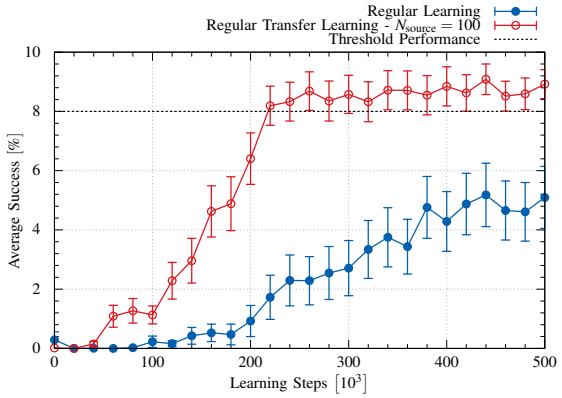


Fig. 3: Comparison of the regular learning process with the fastest transfer learning approach for the deep neural network (DNN). The threshold performance is placed at an average success rate of 80%.

VII. CONCLUSION

In this work, we investigated transfer learning for accelerating learning of mediated interaction tasks from learning experience in unmediated interaction tasks. We found already significant improvements combining a basic transfer learning strategy with a reinforcement learner using Q-learning with linear and non-linear function approximation. Structuring the source task selection with a "scaffolding" strategy led to an additional gain, totalling to a speed-up of almost one order of magnitude for attaining nearly optimal performance in a "extension-of-reach" tool-using task in a simulated 2D world with physics.

We infer from these results that integration of structure into the learning process of the source task is able to further speed up the learning process. Additional studies have shown that the degree of gain depends not only on the used kind of representation but also on the quality of the used baseline. If the learning performance of the agent is excellent from the start, the learning speedup of the proposed learning strategies is much less than applied when e.g. hyperparameters are not well known. While the proposed transfer learning scheme was only evaluated within a simplified but physically realistic

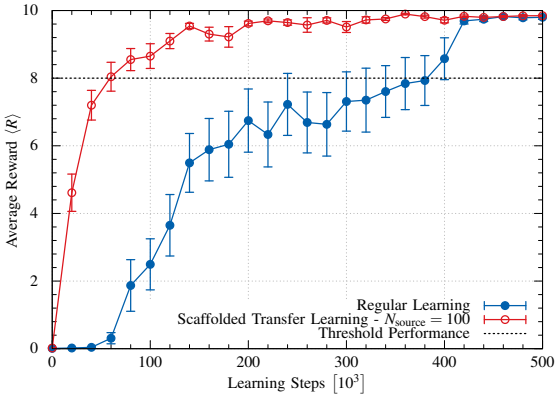


Fig. 4: Comparison of the regular learning process with the fastest transfer learning approach for the RBF representation (RBF). The threshold performance is placed at an average success rate of 80%.

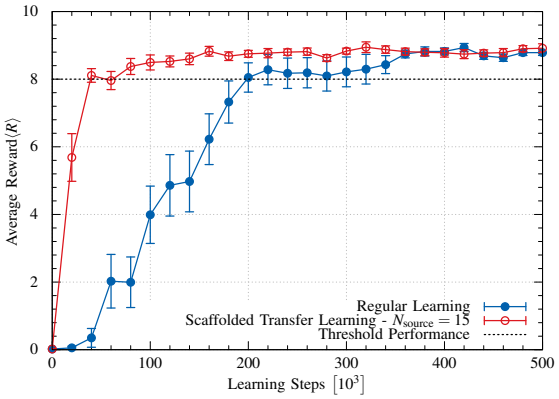


Fig. 5: Comparison of the regular learning process with the fastest transfer learning approach using the binary representation (FSR). The threshold performance is placed at an average success rate of 80%.

toy world, we view the obtained results as encouraging for bringing the general idea to speed up the learning of object interaction scenarios by efficiently pre-learn relevant subskills to more complex 3D domains and also to real-world robot learning tasks.

Still, there are many open questions that might be investigated, as e.g. the impact of the presented or similar transfer learning strategies on learning algorithms different from Q-learning.

ACKNOWLEDGMENT

This research/work was supported by the Cluster of Excellence Cognitive Interaction Technology 'CITEC' (EXC 277) at Bielefeld University, which is funded by the German Research Foundation (DFG).

REFERENCES

[1] E. Visalberghi and L. Trinca, "Tool use in capuchin monkeys: Distinguishing between performing and understanding," *Primates*, vol. 30, no. 4, pp. 511–521, 1989.

[2] A. Stoytchev, "Behavior-grounded representation of tool affordances," in *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*. IEEE, 2005, pp. 3060–3065.

[3] B. Moldovan, P. Moreno, M. van Otterlo, J. Santos-Victor, and L. De Raedt, "Learning relational affordance models for robots in multi-object manipulation tasks," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*. IEEE, 2012, pp. 4373–4378.

[4] L. Jamone, E. Ugur, A. Cangelosi, L. Fadiga, A. Bernardino, J. Piater, and J. Santos-Victor, "Affordances in psychology, neuroscience, and robotics: A survey," *IEEE Transactions on Cognitive and Developmental Systems*, vol. 10, no. 1, pp. 4–25, March 2018.

[5] M. E. Taylor and P. Stone, "Transfer learning for reinforcement learning domains: A survey," *Journal of Machine Learning Research*, vol. 10, no. Jul, pp. 1633–1685, 2009.

[6] G. Konidaris and A. Barto, "Autonomous shaping: Knowledge transfer in reinforcement learning," in *Proceedings of the 23rd international conference on Machine learning*. ACM, 2006, pp. 489–496.

[7] M. E. Taylor, P. Stone, and Y. Liu, "Transfer learning via intertask mappings for temporal difference learning," *Journal of Machine Learning Research*, vol. 8, no. Sep, pp. 2125–2167, 2007.

[8] M. E. Taylor and P. Stone, "Cross-domain transfer for reinforcement learning," in *Proceedings of the 24th international conference on Machine learning*. ACM, 2007, pp. 879–886.

[9] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, "Curriculum learning," in *Proceedings of the 26th annual international conference on machine learning*. ACM, 2009, pp. 41–48.

[10] S. Narvekar, J. Sinapov, M. Leonetti, and P. Stone, "Source task creation for curriculum learning," in *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, 2016, pp. 566–574.

[11] Y. Wu and Y. Tian, "Training agent for first-person shooter game with actor-critic curriculum learning," 2016.

[12] N. Justesen and S. Risi, "Automated curriculum learning by rewarding temporally rare events," *CoRR*, vol. abs/1803.07131, 2018. [Online]. Available: http://arxiv.org/abs/1803.07131

[13] J. Hammond and P. Gibbons, "What is scaffolding?" in *Teachers' voices 8: explicitly supporting reading and writing in the classroom.* Literacy and Numeracy Studies, 2005.

[14] R. S. Sutton and A. G. Barto, *Introduction to Reinforcement Learning*. MIT Press Cambridge, 1998, vol. 135.

[15] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[16] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis, "Mastering the game of Go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, pp. 484–489, 2016. [Online]. Available: http://dx.doi.org/10.1038/nature16961

[17] E. Catto, "Box2d, available from http://www.box2d.org," 2010.

[18] C. J. Watkins and P. Dayan, "Q-learning," *Machine learning*, vol. 8, no. 3-4, pp. 279–292, 1992.

[19] J. Moody and C. J. Darken, "Fast learning in networks of locally-tuned processing units," *Neural computation*, vol. 1, no. 2, pp. 281–294, 1989.

[20] A. Geramifard, T. J. Walsh, S. Tellex, G. Chowdhary, N. Roy, J. P. How *et al.*, "A tutorial on linear function approximators for dynamic programming and reinforcement learning," *Foundations and Trends® in Machine Learning*, vol. 6, no. 4, pp. 375–451, 2013.

[21] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *Journal of Machine Learning Research*, vol. 13, no. Feb, pp. 281–305, 2012.

[22] J. A. Boyan, "Least-squares temporal difference learning," in *ICML*. Citeseer, 1999, pp. 49–56.

[23] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," Dec. 2014.