Noname manuscript No.
(will be inserted by the editor)

Agents Teaching Agents: A Survey on Inter-agent Transfer Learning

Felipe Leno Da Silva · Garrett Warnell · Anna Helena Reali Costa · Peter Stone

Preprint. Get the Journal version at Autonomous Agents and Multi-Agent Systems, 34 (9), 2020, https://doi.org/10.1007/s10458-019-09430-0

Abstract While recent work in reinforcement learning (RL) has led to agents capable of solving increasingly complex tasks, the issue of high sample complexity is still a major concern. This issue has motivated the development of additional techniques that augment RL methods in an attempt to increase task learning speed. In particular, inter-agent teaching – endowing agents with the ability to respond to instructions from others – has been responsible for many of these developments. RL agents that can leverage instruction from a more competent teacher have been shown to be able to learn tasks significantly faster than agents that cannot take advantage of such instruction. That said, the inter-agent teaching paradigm presents many new challenges due to, among other factors, differences between the agents involved in the teaching interaction. As a result, many inter-agent teaching methods work only in restricted settings and have proven difficult to generalize to new domains or scenarios. In this article, we propose two frameworks that provide a comprehensive view of the challenges associated with inter-agent teaching. We highlight state-of-the-art solutions, open problems, prospective applications, and argue that new research in this area should be developed in the context of the proposed frameworks.

Keywords Multiagent Learning · Transfer Learning · Reinforcement Learning

F. L. Silva and A. H. R. Costa
University of São Paulo, Brazil
E-mail: {f.leno,anna.reali}@usp.br

G. Warnell
Army Research Laboratory, USA
E-mail: garrett.a.warnell.civ@mail.mil

P. Stone and F. L. Silva
The University of Texas at Austin, USA
E-mail: pstone@cs.utexas.edu

1 Introduction

Autonomous learning in sequential decision making tasks requires the ability to reason over time-delayed feedback while taking into account environmental, sensory, and actuation stochasticity [36]. In the multiagent setting, the learning process must also contend with the presence and actuation of other actors [8]. Although multiagent variants of reinforcement learning (RL) methods [5] enable learning under such conditions, off-the-shelf RL methods can suffer from high sample complexity, which limits their effectiveness in complex domains. For example, exploring the environment to gather more samples in robotic applications can be dangerous and may cause damage to the robotic platform.

One of the most successful approaches to addressing sample complexity concerns in RL has been that of leveraging the experience of another, more competent agent [51]. Although the literature reports several successful inter-agent teaching strategies in terms of learning speed, real-world applications have to cope with several additional challenges such as differences between the sensors, actuators, and internal representations of the agents involved; development of efficient interfaces and protocols through which instructions will be transferred; and robustness to malicious instructions that could be received from a possibly compromised agent in the system. These challenges are only partially answered (sometimes neglected) in the current body of literature that tends to focus only on single aspects of the teaching process.

In this article, we formulate two inter-agent teaching frameworks: one in which the teacher is responsible for observing the student behavior and initiating the instruction when it is most needed (i.e., *Teacher-Driven*), and one in which the learner is proactive to ask for instructions when desired (i.e, *Learner-Driven*). Although the literature has identified before the importance of who initiates the teaching interaction [1], we present a novel and comprehensive organization and description of all steps involved in those frameworks. We discuss the challenges involved in applying inter-agent transfer methods to complex domains, the state-of-the-art approaches that seek to address these challenges, and the open research questions in the area.

We expect that our contribution will support researchers in the field studying how to leverage inter-agent teaching methods to create complex applications and also those that are currently pursuing open research questions that demand further investigation.

The remainder of this article is organized as follows. Section 2 defines what is the problem we are interested in, delimiting the scope of this article. In Section 3, a brief background on RL is presented, along with the model often used to describe sequential decision problems, in both single-agent and multiagent cases. The detailed description of the two frameworks we propose for inter-agent teaching is in Section 4. In Section 5 we discuss some application examples. Finally, in Section 6 we conclude with remarks on the outcomes of this study.

2 Problem Statement and Scope

An inter-agent teaching relationship requires at least two agents, where a *teacher* agent communicates information to a *learner* – presumably with the intention to accelerate learning (hereafter called *instruction*). We define an instruction as any information communicated by a teacher to a learner with the intention of accelerating learning that *(a)* is specialized to the task at hand, *(b)* can be interpreted and assimilated by the learner, *(c)* is made available during training, and *(d)* is devised without detailed knowledge of the learner's internal representations and parameters (e.g., neural network hyperparameters). Examples of instructions under this definition are *demonstrations* [46], *action advice* [65], and *scalar feedback* [29] on the current learner policy. Importantly, we do **not** consider here information made available before the training process to be an instruction (e.g., a reward shaping [14] or a heuristic function [7] built and made available before learning), nor do we consider as instruction any information that requires explicit knowledge and/or access to the internal models and representations of the learner (e.g., directly copying models from one robotic body to another, or keeping shared models that are directly accessed and updated by multiple agents [39]).

Here, we assume that learner agents are indeed learning, but we do not make the same assumption about teachers, i.e., they may or may not be learning. We further assume that learners are RL agents, but we allow teachers to follow any type of learning algorithm and representation, e.g., teachers may be RL agents, automated agents following a different algorithm, or humans. Finally, we assume that teachers are competent in the learner's task, though they need not be more competent than the learner at all times.

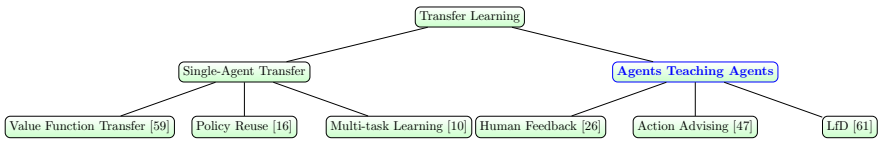

Fig. 1 Depiction of the transfer learning area and the scope of this article. Here, we consider explicitly the subarea of *agents teaching agents* (highlighted in blue). The leaf nodes are representative methods for each transfer learning subarea.

Figure 1 illustrates where the topic considered here fits within the current body of literature. We use the following definition of *transfer learning* as provided by Silva and Costa [49]. In general, RL algorithms map a knowledge space $\mathcal{K}$ to a policy $\pi \in \Pi$, where Π is the space of policies that can be possibly learned by the algorithm [34]. Typically, the knowledge space is composed of samples of interactions with the environment, i.e., $\mathcal{K} = \mathcal{K}^{samples}$. *Transfer learning* [61] consists of leveraging previous knowledge to improve learning speed or performance for a (target) task. This knowledge can come either from a previous (source) task ($\mathcal{K} = \mathcal{K}^{samples} \cup \mathcal{K}^{source}$), or from other

agents ($\mathcal{K} = \mathcal{K}^{samples} \cup \mathcal{K}^{agents}$). Therefore, transfer learning can be divided into two main subareas [51]: the subarea covered by our article, i.e., *agents teaching agents* (ATA)—where knowledge is transferred across agents, and *single-agent transfer* (SA)—where knowledge from source tasks is reused by the same agent. These two subareas can be distinguished by the origin of the knowledge to be reused, as depicted in Table 1. In ATA, the knowledge is generated with respect to target task, but it belongs to another agent. In SA, on the other hand, the learning agent itself generates the knowledge to be reused, but that knowledge is generated with respect to different tasks than the one at hand.

Figure 1 provides an organization of example ATA and SA methods. SA methods include *value function transfer* [62], *policy reuse* [17], and *multi-task learning* [10]. *Learning from feedback* [29], *action advising* [50], and *learning from demonstration* (LfD) [64] are ATA methods to be discussed in this article. For survey articles focused on SA techniques, the reader may refer to the work of Taylor and Stone [61] and Lazaric [34]. Silva and Costa [49] provide a broader survey of both SA and ATA in multiagent tasks, while here we dive deeper into ATA methods and propose general inter-agent teaching frameworks. Argall *et al.* [3] provide a survey focused on learning from demonstration, while we provide a more general view on ATA methods and discuss problems that have been relaxed or neglected by the current literature.

Table 1 Diagram categorizing *single-agent transfer* (SA) and *agents teaching agents* (ATA) according to the origin of the knowledge to be reused. In SA, the knowledge was generated in a different source task by the specific agent that plans to reuse that knowledge. In ATA, the knowledge to be reused was generated in the context of the learner's task, but it was generated by a different agent. If the knowledge was generated by a different agent for a completely unrelated task, there is no reason to reuse it (lower right). If the own agent generates the knowledge specialized for the target task, it corresponds to the typical learning paradigm (upper left).

		Same Agent	
		Y	**N**
Same Task	**Y**	Standard RL	**ATA**
	N	**SA**	Not useful

The practical distinctions between SA and ATA that arise due to differences in knowledge source are made more clear in Table 2. ATA methods require some sort of interface to communicate knowledge from one agent to another since low-level details about the learner are not available to the teacher. However, since the learner has explicit knowledge of its own internal models, such an interface is not necessary for SA techniques. Further, ATA techniques employ instructions that are specialized to and generated for the learner's current task, while SA methods reuse knowledge gathered in while performing a different task.

As a further example of the differences between SA and ATA, consider a line of general-purpose robots that can be trained to perform various tasks. Many

Table 2 Visual representation of the distinctions between *single-agent transfer* (SA) and *agents teaching agents* (ATA). SA methods do not require any interface to transfer knowledge because that knowledge is both acquired a reused by the same agent. ATA techniques, on the other hand, requires an interface to transfer knowledge from one agent to another, i.e., a communication channel (left column). In SA, the agent has explicit knowledge of its own internal representations and models. However, in ATA, the agents involved in knowledge reuse do not have this detailed knowledge of one another (center column). Finally, while the knowledge to be reused for SA is usually generated in the course of addressing a different task and therefore must be adapted to a new one, the knowledge that is used in ATA is specialized to the learner's target task but generated by a different agent (right column).

	Interfacing required	**Internal knowledge**	**Specialized to task**
SA	×	✓	×
ATA	✓	×	✓

tasks will require the robots to be able to execute a set of common behaviors, such as manipulating objects and navigating the environment. Once one of these common behaviors has been learned in the context of one task, it should be the case that it can be reused in order to learn new tasks faster. Similarly, one might imagine copying the memory of one robot to the memory of a new robot that is expected to perform similar tasks such that the new robot can immediately benefit from the knowledge gathered by the first agent. The above types of knowledge reuse are examples of SA.

Now, consider a slightly different scenario in which we deploy one of these robots in an environment where a human is willing to spend some time giving detailed instructions for performing tasks or learning component behaviors. We may also consider cases where another robot – perhaps built by a different manufacturer – is the one available to provide the instruction. In these scenarios, the learner does not have direct access to the internal models and parameters of the prospective teachers and vice versa. This lack of knowledge does not necessarily imply that the instructions are useless, but it does present a new set of challenges to the instruction paradigm, e.g., establishing a transfer protocol such that the agents can understand one another or utilizing knowledge transferred from agents that use different sensors and actuators. These challenges are exactly the ones considered in the study of ATA techniques. Therefore, while both the SA and ATA paradigms share the common objective of knowledge reuse, methods associated with each have to cope with different challenges.

Finally, while we here primarily focus on sequential decision making problems, some of the ideas discussed are applicable to, and have at times been investigated by, other research areas as well, e.g., *active learning* [48].

3 Background

Sequential decision making problems are often modeled as *Markov decision processes* (MDPs) [42]. An MDP is composed of a tuple $\langle S, A, T, R \rangle$, where S is a set of possible states describing both the agent and the environment, A is a

set of actions that can be executed by the agent, $T : S \times A \times S \rightarrow [0, 1]$ is a state transition function where the environment transitions to state $s' \in S$ after the agent applies action $a \in A$ in state $s \in S$ with probability $p(s'|s, a) \sim T$, and $R : S \times A \times S \rightarrow \mathbb{R}$ is a reward function that returns scalar-valued feedback to the agent quantifying the agent's task performance.

In many learning situations, the functions T and R are unknown to the agent. In these situations, agents may utilize *reinforcement learning* (RL) [55] techniques to learn behaviors through interacting with their environment and observing samples of those functions. In particular, after observing the current state s, the agent applies an action a, which leads to a transition to state s' and a reward r. Samples of $\langle s, a, s', r\rangle$ are the only feedback the agent has to learn behaviors that achieve good task performance. More formally, the agent seeks to learn an optimal policy $\pi^* : S \rightarrow \Delta(A)$ (where $\pi^*(a|s)$ denotes the likelihood of the policy selecting action a while in state s) that dictates, for any state $s \in S$, the output distribution over actions will lead to maximization of the expected sum of discounted future rewards. A policy π that approximates π^* can be learned from samples using many RL techniques. Broadly speaking, however, there are two classes of approach. In the first class, the agent attempts to directly learn a parameterized policy π_θ by iteratively updating θ so that π_θ gets as close as possible to π^* [56]. In the second class of RL approaches, the agent tries to learn a value function $Q : S \times A \rightarrow \mathbb{R}$ that estimates the expected sum of discounted futures rewards for state-action pairs, and then uses Q to set the policy by selecting the action with highest value estimate [67].

MDPs generalize to the multiagent case as *stochastic games* (SG) [9]. A SG is a tuple $\langle S, U, T, R_1, \ldots R_n\rangle$, where n denotes the number of agents present in the environment, S is the state space comprised of the local state of all agents $S = S_1 \times \ldots \times S_n$, and $U = A_1 \times \ldots \times A_n$ is the joint action space (each agent chooses one action from their respective action sets simultaneously at each step). In SGs, the transition function now depends on the *joint* state-action information, i.e., $T : S \times U \times S \rightarrow [0, 1]$, and every agent has its own reward function. Since each agent does not necessarily pursue the same objective, SGs are typically solved using scenario-specific algorithms [15, 26, 33]. That said, these algorithms typically follow the same general idea as those for single-agent learning, i.e., they either directly approximate the joint policy or joint value functions.

The main challenge of applying RL in both MDPs and SGs is that—especially for more classical algorithms—learning even seemingly-simple tasks can require a large amount of experience (i.e., large sample complexity). For this reason, the research community has also been investigating ways in which the RL paradigm can be improved, especially for more complex applications. One such way is to consider scenarios in which the learner receives instructions from another, more-experienced agent. In both MDP and SG settings, humans and/or other artificial agents are sometimes able to provide various forms of instruction that can help a learning agent build good initial policies, disambiguate knowledge, and/or reduce the amount of experience required in

order to learn an acceptable policy [49]. Designing a framework that allows for agents to instruct one another, i.e., *inter-agent teaching*, requires dealing with a number of challenges. For example, one must design appropriate inter-agent communication protocols, workable interfaces that consider differences in agent sensors and actuators, and reasonable strategies for translating knowledge if agents use different internal representations.

The majority of work in the literature related to agents teaching agents focuses on only a portion of those problems—none has outlined and discussed completely the modules that must be combined in order to design an efficient and effective framework. To the best of our knowledge, this is the first article that provides a comprehensive view of inter-agent teaching methods.

4 Proposed Frameworks

We shall now describe our own perspective on inter-agent teaching algorithms and implementations. We start by providing high-level descriptions of two broad categories of ATA techniques: those that are *learner-driven*, and those that are *teacher-driven*. We then detail specific components of each of these frameworks and discuss possible implementations based on the current body of literature.

Broadly speaking, we categorize ATA techniques into one of the two frameworks depicted in Figures 2 and 3, i.e., learner-driven and teacher-driven interactions. Figure 2 depicts the learner-driven framework, in which the *learner* is responsible for initiating the interaction between agents. Under this framework, the learning agent must first *generate a behavior*, i.e., attempt to perform its task using some initial policy. Then, throughout the course of the learning process, it is up to the learning agent to decide when and how to *define a query* to send to a (potential) teacher. Assuming the query is successfully received, the teaching agent then *evaluates the utility* of actually providing instruction to the learner in the context of the current situation. If the teacher deems the situation worthy of instruction, the teacher then *defines the instruction* and communicates that instruction to the learner. Finally, the learner then *updates its knowledge* in response to the instruction, after which it is ready to initiate another interaction with the teacher and/or resume learning through its own means.

In contrast to the learner-driven framework for ATA, Figure 3 depicts the teacher-driven framework, in which the *teacher* initiates the interaction between agents. The main difference between this framework and the learner-driven framework is that, in this configuration, there is no explicit query generated by the learner. This lack of query means that it is up to the teacher to decide when the instruction takes place. Although this is a relatively small change at a high level, at the implementation level, it means that the teacher's utility-evaluation module has to cope with new challenges regarding when to define and communicate instruction.

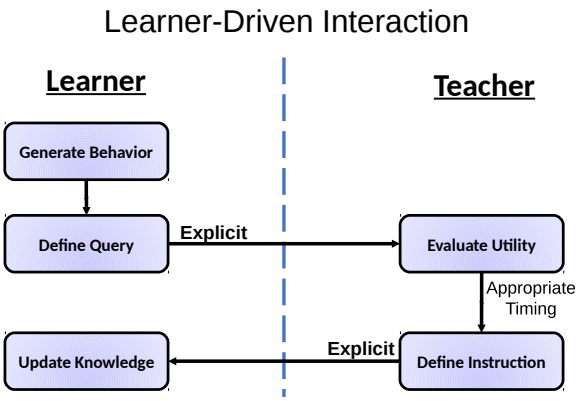

Fig. 2 A graphical depiction of the learner-driven ATA framework. The dashed line separates the learning agent from the teaching agent, where it is only possible to transmit information from one agent to another using either implicit or explicit communication, depicted using an arrow. Rectangles correspond to inter-agent teaching modules, which are described in Section 4.1 to 4.5.

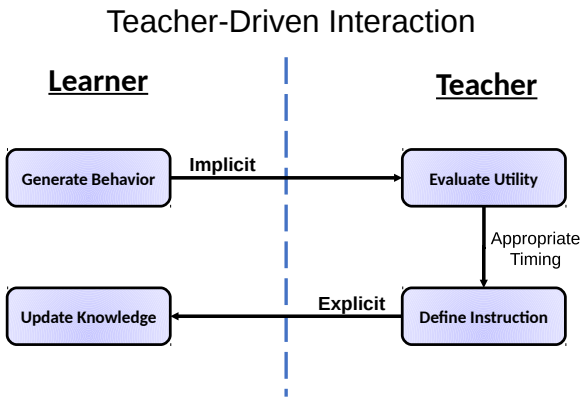

Fig. 3 A graphical depiction of the teacher-driven ATA framework. The dashed line separates the learning agent from the teaching agent, where it is only possible to transmit information from one agent to another using either implicit or explicit communication, depicted using an arrow. Rectangles correspond to inter-agent teaching modules, which are described in Sections 4.1 to 4.5.

In the following subsections, we detail each of the component modules defined above, list the main problems each module has to cope with, and discuss state-of-the-art solutions. Table 3 summarizes the modules to be described.

Table 3 List of all inter-agent teaching modules and the challenges they have to cope with. Modules that are exclusively for learner-driven approaches are marked with †, and modules exclusively for teacher-driven approaches are marked with ‡. When available, references to representative works addressing each of the challenges are given (or to a survey when many of them exist).

Behavior Generation (Sec. 4.1)	**Instruction Definition (Sec. 4.4)**
Query Definition (Sec. 4.2)†	- Instruction construction [49]
- Query timing [11,28,35,40,50]	- Interfacing and translating instruction
- Teacher selection	**Knowledge Update (Sec. 4.5)**
- Query construction	- Receiving instruction
Utility Evaluation (Sec. 4.3)	- Instruction reliability
- Behavior observation‡ [1,41]	- Knowledge merging [49]
- Instruction timing [40,41,50]	

4.1 Behavior Generation

Throughout the learning process, the learner must maintain some sort of method of behavior generation, i.e., a policy. It is in the context of this behavior generation that each inter-agent teaching interaction takes place. To start, before any learning can take place, the learner must first generate an initial behavior from which it can start exploring.[1] Generally, RL agents use a random policy, though perhaps better initial policies can also be found using the agents own experiences from similar previous tasks [61] (i.e., SA). Once the initial policy has been set, the inter-agent interaction can begin, i.e., the agent can then either generate its query (in the learner-driven paradigm) or wait for instructions (in the teacher-driven interaction). Once the inter-agent interaction has taken place and the learner has updated its knowledge and behavior generator, the inter-agent interaction can begin anew.

4.2 Query Definition

In the context of its current behavior, the agent must define *when*, *to whom*, and *how* to ask for instruction. Answering each of these questions poses an interesting research problem, and solutions have been implemented in different ways depending on the particular assumptions made and applications considered. This problem is relevant for learner-driven approaches.

- **Query timing** – *When* the learner should query the teacher is important for two main reasons:
  1. **Communication is a scarce resource in many multiagent applications**: In principle, it is possible to receive instructions at every time step of the agent's learning process [58]. In many applications, though, communication is limited by, e.g., available bandwidth [53] or battery

[1] Notice that the initial policy definition might be implicit (e.g., assuming the agent starts with a random policy), effectively enabling the inter-agent interaction to start immediately at the beginning of the learning process.

power (e.g., when the agents are standalone robots or wireless sensors). In many works, communication scarcity is modeled through a *budget* (i.e., maximum amount of inter-agent interactions) [65, 69, 1, 50], imposing a hard threshold on how much agents are able to communicate. In the case of human teachers in particular, communication is also limited by human factors such as attention span and perceived tediousness of instruction due to excessive interaction requests. Therefore, in general, it is desirable for inter-agent teaching systems to limit the number of queries to only those that are most needed. One very common procedure adopted by most *learning from demonstration* methods [3] is to query for instructions at fixed, predefined time, such as asking for a fixed demonstrations at the very beginning of the learning procedure [45, 64]. Another option is to query randomly according to a decaying probability that eventually approaches zero [12]. Those procedures both restrict the required communication overhead to only that which is incurred during a known, short period of time, and allow the agent to receive instruction from teachers that will not be available in the system for a long period of time (e.g., when humans provide instructions for a certain period of time and then leave the system). However, such querying procedures are not efficient in the use of instructions, because the instructions themselves may be sub-optimal, e.g., they may be given for situations in which the learner already has a good policy [69].
Another possible querying strategy that can reduce communication overhead is for the learner to estimate its confidence in its own policy, and to query the teacher only in situations where that confidence is low [1, 50]. Since estimating confidence accurately is sometimes challenging, the possibility of learning a *student policy* has also been explored, i.e. modeling *when* to ask for instructions as a second task to be learned simultaneously with the actual task [40, 70]. However, adding a second learning task might slow overall learning, and this approach has only been shown to work well in simple tasks. Whether this idea can scale to a broad range of complex domains or not is still an open question.

2. **Instructions might hamper the learning process in some situations**: Although teachers are expected to perform well at the learner's task, it may happen that the learner has acquired an even better policy for some situations [69]. Therefore, if the learner were to query the teacher arbitrarily, an extra burden might be placed on the *knowledge update* module (Section 4.5) to identify when the instruction might result in worse behavior than if it were not given at all. This concern is especially relevant to situations in which the teachers themselves are also learning the task [40, 50]. Ideally, the learner would be able to estimate portions of its policy that still need improvement [35] and query for instructions directed toward those portions [11, 28].

– **Teacher selection** – After the learner determines when to query a prospective teacher, it might have to reason about *whom* to query. Most inter-agent

teaching methods assume that the teacher is known and has agreed to provide instructions. Even in these cases, practical issues might dictate the teacher is inaccessible for portions of the learning process (e.g., the human teacher is not available at that time or an automated teacher is too far away to communicate with through a wireless network). Adaptive teacher-definition algorithms have not been the subject of extensive research, and how to automatically identify, engage with, and estimate the trustworthiness of a new teacher is an open area of research. When multiple agents could assume the role of the teacher [69], a proper teacher selection strategy becomes even more important. However, most existing literature assumes that all possible teachers can always be reached, which is not true for all applications [50,69]. Taylor *et al.* [59] use the concept of a neighborhood, where agents in close physical proximity are able to communicate to each other. However, the neighborhood is both manually defined and static—how to adaptively include or remove agents from neighborhoods is still an open problem.

– **Query construction** – After the questions of when to query and whom to query have been answered, the task of *constructing* the query may have to be considered. Again, this is itself a challenging research problem, involving both adhering to a given query protocol, and deciding what information should be transmitted as part of the query. Most existing work implicitly assumes that the protocols for querying and receiving instructions are predefined and known by all agents, which may not be a valid assumption in many real-world applications [49,51]. Kono *et al.* [31] proposed the construction of a web ontology to translate particularities of each robot (e.g., low-level detail pertaining to sensors and actuators) into a common, abstract description that could be understood by any agent with access to the ontology. A similar idea could be applied for ATA query protocol definition, though the task of manually building and maintaining such an ontology would be challenging. *Learning* an inter-agent query construction protocol may also be possible. The literature has only recently started exploring how agents can learn to communicate between themselves [19,23, 54], and these techniques could be adapted to specific ATA query construction protocols. Another area to be explored is the use of natural language for such communication, facilitating communication mainly when there is interaction of automated agents with humans.
Given the protocol, the second task of query construction is to define what to transmit in the query. If the teacher is able to deduce all the information it needs to give the instruction directly from observation of the learning agent, then the query might simply be a message that, under the communication protocol, corresponds to an instruction request. Alternatively, the learner can communicate additional information to help the teacher [35], such as its current observations [50], its intended action for the current state [60], its uncertainty in the current state [35], or more complex information. For example, In Cui and Niekum's work [13], the agent transfers

a trajectory of actions to the teacher, and the teacher indicates which of the chosen actions were bad.

4.3 Utility Evaluation

Under both the teacher-driven and learner-driven frameworks, an important component of the interaction is the strategy that the teacher uses to decide whether or not instructions should be provided to the learner, i.e., *utility evaluation*. We outline here two fundamental concerns in utility evaluation: behavior observation and instruction timing.

- **Behavior observation** – For teacher-driven approaches, deciding when to *observe* the learner's behavior is of utmost importance. While many methods assume that the teacher will observe the learner during the entire training process [41,60,65], constant observation is impractical in many situations. When teachers are artificial agents, the communication costs of making observations at every step might be prohibitive (e.g., excessive power needed to operate visual sensors). When teachers are humans, constant observation would require long periods of availability and alertness. In addition to having limited attention spans [1], humans might also tire of observing the learner, and the resulting lack of engagement can lead to low-quality instruction [35]. One straightforward way to address some of these problems is to do away with learner observation altogether (e.g., provide a preset number of instructions at the beginning of learning [3]). However, adopting this strategy means that the teacher is not able to give instructions tailored to the learner.

- **Instruction timing** – For both teacher-driven and learner-driven approaches, a second fundamental concern in utility evaluation deciding when to send the instruction. However, it is not easy to decide when and what kind of instruction might be useful to the learner. One possible solution is to endow the learner with the ability to modify its behavior to indicate when instructions are most needed, e.g., slowing down its actuation when the confidence in its policy is low [41]. Another issue that is important to consider is reaction time, i.e., the amount of time it takes for the teacher to process the observation and provide instruction. This is especially important in situations where the teacher is a human [25,29,37].

4.4 Instruction Definition

After determining that the current state is appropriate for giving an instruction, the question now is how to define and represent the instruction to be transferred. This is especially challenging if agents have different or unknown representations, or different sensors and actuators, requiring some kind of interface or translation to enable communicating the instruction successfully.

– **Instruction construction** – Several types of instruction have been studied in the literature. Humans can provide *action advice* to automated learners, where the teacher suggests actions that the learner should take. One form of action advice is communicated through *rules* that prescribe, in particular scenarios, that either particular actions should be taken or that certain actions should be preferred over others [38]. While this form of instruction has proven successful for some domains and behaviors, determining precisely which rules will lead to the desired behavior can prove challenging for a human teacher. A more flexible form of instruction that could be interpreted as dense action advice is that of *demonstration* [3]. Using this paradigm, teachers typically provide instruction by attempting to teleoperate the learner [44], applying external forces to the learner so as to generate a kinesthetic demonstration [24], or by simply performing an instructive behavior for the learner to observe [47]. While instructing learners through demonstration has been successful for many tasks, it may sometimes prove difficult or impossible for the teacher to provide a demonstration of the task. Moreover, in cases where the teacher performs the instructive behavior for the learner to observe, it can be difficult for the agent to overcome factors such as embodiment mismatch and lack of complete observability [22]. Human teachers can also provide instruction to the learner through natural language [32]. Providing language instructions is far less burdensome to the teacher, but comes with the requirement that the learner be able to map the language to portions of its state and action space, and the ability to do so can come with significant training time. Providing *preferences* [68] as instructions is also possible and is inter-operable with preference-based RL algorithms. Finally, there has recently been interest in the research community on allowing human teachers to communicate with automated learners using *feedback* [29,37,66], i.e., scalar values or preferences communicated to the agent. This paradigm is the least burdensome of those discussed here, requiring only that the teacher observe learner behavior and provide a numerical score or an ordering of that behavior.
For situations in which the teacher is also an automated agent, the most commonly applied instruction type is *action advice* [16,40,50,60,65]. In principle, any kind of instruction given by humans would also be applicable to be used between automated agents. However, if the agents have a common understanding of the actions and access to state observation, communicating actions is easier and more flexible than trying to translate the internal models of an agent to other types of human-like feedback.
Importantly, there has been relatively little work that has studied simultaneously using *multiple* instruction types [49]. Given the strengths and weaknesses of the instruction styles discussed above, studying paradigms that allow teachers to utilize combinations of instruction types could provide a way to mitigate individual weaknesses and amplify the amount of instructive signal that the learner can ultimately use to refine its behavior.
The impact of the learner actuation on the instruction to be later provided is an important yet little studied aspect of this module. Although the

learner's policy does not normally affect instruction construction in inter-agent relations between automated agents, humans might (consciously or not) adapt their instructions to the observed behavior, and some work has indeed shown it to be an advantage to explicitly model human feedback as dependent upon the agent's current behavior [37]. Further investigations could explore how to adapt instructions to correct undesired behaviors, possibly placing importance on the *Behavior observation* challenge for learner-driven approaches as well.

– **Interfacing and translating instruction** – The interface by which the teacher communicates its instructions is also an important factor to consider in all teaching frameworks. Such interfaces consist of two critical components: *(a)* the way in which observations of the learner are presented to the teacher, and *(b)* the way in which instructions are presented to the learner.
In most cases, provided that the agents have previously agreed on a communication protocol, the interface needed for instruction between automated agents is a communication channel. Then, the teacher might be able to directly observe the learner's state through its own sensors or, if this is not possible, the learner transfers its observations through the communication channel.
Considering human teachers, observations of the learner are of course limited to the typical sensing capabilities of a human – there is as of yet no truly direct way in which an automated agent can communicate low-level state information to a human. In the case of physical automated agents (e.g., robots), a human teacher may view the agent as they would anything else in the world and is forced to infer the learner's state information from this experience. In the case of virtual automated agents, the virtual learner's state is typically communicated via visualization on a computer monitor. For example, video-game playing agents often present a human teacher with the same game visualization that humans would use to play the game themselves [6], and physics-based simulators present the teacher with video renderings of the environment from a particular vantage point [63]. The interfaces by which human teachers communicate instruction to learners are also varied. Rule-based instruction can be provided to the agent by a computer programming interface. Instruction that comes in the form of demonstration is typically communicated by joystick or keyboard, though, in the case of kinesthetic demonstration, it can also be communicated by physically interacting with the learner. Natural-language instruction can be communicated from the human to the learner through a microphone (for spoken language) or keyboard (for written language). Feedback instructions are typically communicated through button presses, i.e., the human teacher presses one button to communicate that the learner's current behavior is "good", and another to communicate that the teacher thinks the learner's behavior is "bad", though there has recently been work that has

inferred feedback values by analyzing a video stream of the human's facial expression and trying to estimate emotional state [2].
While various interfaces have been used in the teacher-learner paradigm, understanding the impact of interface design choices on the learning process itself is an open problem. In the context of human teachers in particular, poor interface choices have the potential to drastically reduce the quality of instruction – if the human teacher cannot infer the relevant learner state information, it may be impossible to know what instruction is appropriate. With respect to robotic learners in particular, there recently has been work on providing visualization of hidden internal state information through means such as novel light displays [18,57] and augmented reality [43], but, unfortunately, none of these visualization techniques has yet been studied in the context of the inter-agent teaching.

4.5 Knowledge Update

Finally, after the teacher issues an instruction, the learner is faced with the problem of updating its own knowledge using the information contained in that instruction. Major components of this problem for both learner-driven and teacher-driven approaches include *receiving* the instruction, determining the *reliability* of the instruction, and *merging* the instruction with the learner's existing knowledge.

- **Receiving instruction** – The first step in the knowledge update process is for the learner to successfully receive the instruction. Most of the ATA literature assumes that all teachers will answer when queried, and also that all instructions will be correctly received by the learner. In practical applications, however, the query or the instruction might be corrupted or lost due to, e.g., a faulty communication channel. No method in the literature explicitly considers this problem, and doing so is a promising research opportunity.

- **Instruction reliability** – After successfully receiving the instruction, the learner should decide whether or not it is reliable. As with receiving instruction, most of the existing ATA literature simply assumes that all instructions are reliable, and proceeds immediately to the *knowledge merging* step. However, estimating the reliability of an instruction is of utmost importance in applications where teachers can be compromised, hacked, or might have malicious intentions. Even when we can assume that all teachers are benevolent, teachers might have a worse policy than the learner in some situations, and therefore give bad instructions that should be discarded. One possible solution is to place the burden on the design of a reliable confidence function [1,50], which could allow agents to compare expected performance. Other than this, to the best of our knowledge, no other efficient solutions to this problem exist, and how to perform instruction reliability determination effectively is still an open problem. Possible

solutions could be the development of a trust framework that would allow learners to estimate the quality of instruction, or requiring instructions to also come with explanations that the learner could examine for validity.

– **Knowledge merging** – The final step in the knowledge update component is for the learner to merge the received instruction with its current knowledge. This step is highly dependent on which kind of information was made available in the instruction (Section 4.4), and it can range from performing exploration according to what was suggested in the instruction [1, 50, 60, 65], to performing more classical knowledge update operations using samples gathered according to the instruction, to learning entirely different models using scalar feedback given as instructions [29].

5 Application Examples

Inter-agent teaching methods can potentially be applied to any RL domain and have successfully been used to solve complex tasks such as video game playing [60], robotics [11], and smart grid scenarios [59]. However, real-world domains that have at least one of the following characteristics are most likely to directly benefit from the techniques discussed in this article:

1. Exploring the environment is dangerous or expensive enough to make it worthwhile to use a human instructor;
2. Learning the task alone is difficult due to the size of the state space and/or reward sparsity;
3. The task itself is a multiagent problem in which the configuration of agents might possibly change over time (or new agents might appear during learning) and internal details of other agents are not available;
4. Access to good baseline behavior is readily available from handcrafted strategies or any other learning algorithms.

Robotics applications are the most present real-world application in the inter-agent teaching literature [30], primarily because they exhibit a combination of characteristics 1 and 2. Robotic equipment is expensive and fragile, hence it might be too dangerous to learn directly from exploration even in controlled environments. Furthermore, usually the agent states are derived from continuous state features, which makes the efficient exploration of the state space very challenging. As robots start to become more integrated in our daily lives, characteristic 3 will apply as well – robots from different manufacturers will be potentially sharing the same environment, and they could give instructions to one other.

The ad hoc teamwork research area [52] focuses specifically on domains that have characteristic 3. For this reason, domains from this area are also promising application areas for inter-agent teaching. Robot soccer is one prominent domain that has been consistently explored by ad hoc teamwork methods [4].

Finally, domains that have been historically approached with fixed, handcoded policies and for which learning agents are starting to be employed are

likely to benefit from inter-agent teaching methods because of characteristic 4. For example, hand-coded or rule-based medical treatment policies [27] have been widely applied. In these domains, RL agents cannot perform random exploration because each action corresponds to an effect – perhaps one that is unsafe – on patient health. However, learning agents in these domains could benefit from interaction with hand-coded or older systems that are known to have a reasonably safe policy. Moreover, improving upon those policies might help to alleviate some of the challenges that are currently hampering the adoption of RL to this domain, such as the possible mismatch between actions proposed by the RL algorithm and the actually performed procedure when learning from historical data [20, 21].

6 Conclusion

Inter-agent teaching methods have played an important role in trying to augment RL methods to increase task learning speed. However, existing literature presents solutions for only some of the many challenges involved in designing these inter-agent methods. In this article, we have provided a comprehensive view of these challenges, and also outlined two broad categories in which to organize them, i.e., *learner-driven* and *teacher-driven* methods. We have also discussed the state-of-the-art options available to implement various modules required in these frameworks, and we have highlighted open research questions and promising application domains. We argue that inter-agent teaching methods should be designed with the global view we have provided in mind, and we hope that our contributions will help to bridge the gap between the current state-of-the-art inter-agent teaching methods and real-world applications where RL agents have not yet been applied.

Acknowledgements

This work has taken place in the Learning Agents Research Group (LARG) at UT Austin. LARG research is supported in part by NSF (IIS-1637736, IIS-1651089, IIS-1724157), ONR (N00014-18-2243), FLI (RFP2-000), ARL, DARPA, Intel, Raytheon, and Lockheed Martin. Peter Stone serves on the Board of Directors of Cogitai, Inc. The terms of this arrangement have been reviewed and approved by the University of Texas at Austin in accordance with its policy on objectivity in research. We also gratefully acknowledge financial support from CNPq, grants 425860/2016-7 and 307027/2017-1 and São Paulo Research Foundation (FAPESP), grants 2015/16310-4 and 2018/00344-5.

References

1. Amir, O., Kamar, E., Kolobov, A., Grosz, B.: Interactive Teaching Strategies for Agent Training. In: Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI), pp. 804–811 (2016)

2. Arakawa, R., Kobayashi, S., Unno, Y., Tsuboi, Y., Maeda, S.i.: DQN-TAMER: Human-in-the-loop Reinforcement Learning with Intractable Feedback. arXiv preprint arXiv:1810.11748 (2018)
3. Argall, B.D., Chernova, S., Veloso, M., Browning, B.: A Survey of Robot Learning from Demonstration. Robotics and Autonomous Systems **57**(5), 469 – 483 (2009). DOI https://doi.org/10.1016/j.robot.2008.10.024
4. Barrett, S., Stone, P.: Cooperating with Unknown Teammates in Complex Domains: A Robot Soccer Case Study of Ad Hoc Teamwork. In: Proceedings of the 29th AAAI Conference on Artificial Intelligence (AAAI), pp. 2010–2016 (2015)
5. Bazzan, A.L.C.: Beyond Reinforcement Learning and Local View in Multiagent Systems. Künstliche Intelligenz **28**(3), 179–189 (2014). DOI 10.1007/s13218-014-0312-5
6. Bellemare, M.G., Naddaf, Y., Veness, J., Bowling, M.: The Arcade Learning Environment: An Evaluation Platform for General Agents. Journal of Artificial Intelligence Research (JAIR) **47**, 253–279 (2013)
7. Bianchi, R.A.C., Martins, M.F., Ribeiro, C.H.C., Costa, A.H.R.: Heuristically-Accelerated Multiagent Reinforcement Learning. IEEE Transactions on Cybernetics **44**(2), 252 – 265 (2014). DOI 10.1109/TCYB.2013.2253094
8. Bowling, M., Veloso, M.: An Analysis of Stochastic Game Theory for Multiagent Reinforcement Learning. Tech. rep., Computer Science Department, Carnegie Mellon University (2000)
9. Busoniu, L., Babuska, R., De Schutter, B.: A Comprehensive Survey of Multiagent Reinforcement Learning. IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews **38**(2), 156–172 (2008). DOI 10.1109/TSMCC.2007.913919
10. Calandriello, D., Lazaric, A., Restelli, M.: Sparse Multi-Task Reinforcement Learning. In: Z. Ghahramani, M. Welling, C. Cortes, N.D. Lawrence, K.Q. Weinberger (eds.) Advances in Neural Information Processing Systems (NIPS), pp. 819–827. Curran Associates, Inc. (2014). URL http://papers.nips.cc/paper/5247-sparse-multi-task-reinforcement-learning.pdf
11. Chernova, S., Veloso, M.: Interactive Policy Learning through Confidence-Based Autonomy. Journal of Artificial Intelligence Research (JAIR) **34**(1), 1–25 (2009)
12. Clouse, J.A.: Learning from an automated training agent. In: Adaptation and Learning in Multiagent Systems. Springer Verlag (1996)
13. Cui, Y., Niekum, S.: Active Reward Learning from Critiques. In: IEEE International Conference on Robotics and Automation (ICRA), pp. 6907–6914 (2018)
14. Devlin, S.: Potential-Based Reward Shaping for Knowledge-Based, Multi-Agent Reinforcement Learning. Ph.D. thesis, University of York (2013)
15. Dusparic, I., Harris, C., Marinescu, A., Cahill, V., Clarke, S.: Multi-Agent Residential Demand Response Based on Load Forecasting. In: 1st IEEE Conference on Technologies for Sustainability (SusTech), pp. 90–96 (2013). DOI 10.1109/SusTech.2013.6617303
16. Fachantidis, A., Taylor, M.E., Vlahavas, I.: Learning to Teach Reinforcement Learning Agents. Machine Learning and Knowledge Extraction **1**(1) (2018). DOI 10.3390/make1010002
17. Fernández, F., Veloso, M.: Probabilistic Policy Reuse in a Reinforcement Learning Agent. In: Proceedings of the 5th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS), pp. 720–727 (2006). DOI 10.1145/1160633.1160762
18. Fernandez, R., John, N., Kirmani, S., Hart, J., Sinapov, J., Stone, P.: Passive Demonstrations of Light-Based Robot Signals for Improved Human Interpretability. In: IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN) (2018)
19. Foerster, J.N., Assael, Y.M., de Freitas, N., Whiteson, S.: Learning to Communicate with Deep Multi-Agent Reinforcement Learning. In: Conference on Neural Information Processing Systems (NIPS) (2016)
20. Gottesman, O., Johansson, F., Komorowski, M., Faisal, A., Sontag, D., Doshi-Velez, F., Celi, L.: Guidelines for Reinforcement Learning in Healthcare. Nature Medicine **25**, 16–18 (2019)
21. Gottesman, O., Johansson, F.D., Meier, J., Dent, J., Lee, D., Srinivasan, S., Zhang, L., Ding, Y., Wihl, D., Peng, X., Yao, J., Lage, I., Mosch, C., Lehman, L.H., Komorowski,

M., Faisal, A., Celi, L.A., Sontag, D., Doshi-Velez, F.: Evaluating Reinforcement Learning Algorithms in Observational Health Settings. arXiv preprint **abs/1805.12298** (2018). URL http://arxiv.org/abs/1805.12298
22. Gupta, A., Devin, C., Liu, Y., Abbeel, P., Levine, S.: Learning Invariant Feature Spaces to Transfer Skills with Reinforcement Learning. In: Proceedings of the 5th International Conference on Learning Representations (ICLR) (2017)
23. Hausknecht, M., Stone, P.: Grounded Semantic Networks for Learning Shared Communication Protocols. In: NIPS Workshop on Deep Reinforcement Learning (2016)
24. Hersch, M., Guenter, F., Calinon, S., Billard, A.: Dynamical System Modulation for Robot Learning via Kinesthetic Demonstrations. IEEE Transactions on Robotics **24**(6), 1463–1467 (2008)
25. Hockley, W.E.: Analysis of Response Time Distributions in the Study of Cognitive Processes. Journal of Experimental Psychology: Learning, Memory, and Cognition **10**(4), 598 (1984)
26. Hu, Y., Gao, Y., An, B.: Multiagent Reinforcement Learning with Unshared Value Functions. IEEE Transactions on Cybernetics **45**(4), 647–662 (2015)
27. Jonsson, A.: Deep Reinforcement Learning in Medicine. Kidney Diseases **5**(1), 3–7 (2019)
28. Judah, K., Fern, A.P., Dietterich, T.G., Tadepalii, P.: Active Imitation Learning: Formal and Practical Reductions to I.I.D. Learning. Journal of Machine Learning Research (JMLR) **15**(1), 3925–3963 (2014)
29. Knox, W.B., Stone, P.: Interactively Shaping Agents via Human Reinforcement: The TAMER Framework. In: Proceedings of the 5th International Conference on Knowledge Capture, pp. 9–16 (2009)
30. Kober, J., Bagnell, J.A., Peters, J.: Reinforcement Learning in Robotics: A Survey. The International Journal of Robotics Research **32**(11), 1238–1274 (2013). DOI 10.1177/0278364913495721
31. Kono, H., Kamimura, A., Tomita, K., Murata, Y., Suzuki, T.: Transfer Learning Method Using Ontology for Heterogeneous Multi-agent Reinforcement Learning. International Journal of Advanced Computer Science and Applications (IJACSA) **5**(10), 156–164 (2014). DOI 10.14569/IJACSA.2014.051022
32. Kuhlmann, G., Stone, P., Mooney, R., Shavlik, J.: Guiding a Reinforcement Learner with Natural Language Advice: Initial Results in RoboCup Soccer. In: AAAI Workshop on Supervisory Control of Learning and Adaptive Systems (2004)
33. Lauer, M., Riedmiller, M.: An Algorithm for Distributed Reinforcement Learning in Cooperative Multi-Agent Systems. In: Proceedings of the 17th International Conference on Machine Learning (ICML), pp. 535–542 (2000)
34. Lazaric, A.: Transfer in Reinforcement Learning: A Framework and a Survey, pp. 143–173. Springer Berlin Heidelberg, Berlin, Heidelberg (2012)
35. Li, G., Hung, H., Whiteson, S., Knox, W.B.: Using Informative Behavior to Increase Engagement in the TAMER Framework. In: Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems (AAMAS), pp. 909–916 (2013)
36. Littman, M.L.: Reinforcement Learning Improves Behaviour from Evaluative Feedback. Nature **521**(7553), 445–451 (2015). DOI 10.1038/nature14540
37. MacGlashan, J., Ho, M.K., Loftin, R., Peng, B., Wang, G., Roberts, D.L., Taylor, M.E., Littman, M.L.: Interactive Learning from Policy-Dependent Human Feedback. In: Proceedings of the 34th International Conference on Machine Learning (ICML), pp. 2285–2294 (2017)
38. Maclin, R., Shavlik, J., Torrey, L., Walker, T., Wild, E.: Giving Advice about Preferred Actions to Reinforcement Learners via Knowledge-based Kernel Regression. In: Proceedings of the 20th AAAI Conference on Artificial Intelligence (2005)
39. Mnih, V., Badia, A.P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., Silver, D., Kavukcuoglu, K.: Asynchronous Methods for Deep Reinforcement Learning. In: Proceedings of the 33rd International Conference on Machine Learning (ICML), pp. 1928–1937 (2016)
40. Omidshafiei, S., Kim, D., Liu, M., Tesauro, G., Riemer, M., Amato, C., Campbell, M., How, J.P.: Learning to Teach in Cooperative Multiagent Reinforcement Learning. In: Proceedings of the 33rd AAAI Conference on Artificial Intelligence (AAAI) (2019)

41. Peng, B., MacGlashan, J., Loftin, R., Littman, M.L., Roberts, D.L., Taylor, M.E.: A Need for Speed: Adapting Agent Action Speed to Improve Task Learning from Non-Expert Humans. In: Proceedings of the 15th International Conference on Autonomous Agents and Multiagent Systems (AAMAS), pp. 957–965 (2016)
42. Puterman, M.L.: Markov Decision Processes : Discrete Stochastic Dynamic Programming. J. Wiley & Sons, Hoboken (N. J.) (2005)
43. Reardon, C., Lee, K., Fink, J.: Come See This! Augmented Reality to Enable Human-robot Cooperative Search. In: IEEE International Symposium on Safety, Security, and Rescue Robotics (2018)
44. Ross, S., Melik-Barkhudarov, N., Shankar, K.S., Wendel, A., Dey, D., Bagnell, J.A., Hebert, M.: Learning Monocular Reactive UAV Control in Cluttered Natural Environments. In: IEEE International Conference on Robotics and Automation (ICRA) (2013)
45. Santara, A., Naik, A., Ravindran, B., Das, D., Mudigere, D., Avancha, S., Kaul, B.: RAIL: Risk-Averse Imitation Learning. In: Proceedings of the 17th International Conference on Autonomous Agents and Multiagent Systems (AAMAS), pp. 2062–2063 (2018)
46. Schaal, S.: Learning from Demonstration. In: Advances in Neural Information Processing Systems (NIPS), pp. 1040–1046 (1997)
47. Sermanet, P., Lynch, C., Chebotar, Y., Hsu, J., Jang, E., Schaal, S., Levine, S., Brain, G.: Time-Contrastive Networks: Self-supervised Learning from Video. In: IEEE International Conference on Robotics and Automation (ICRA) (2018)
48. Settles, B.: Active Learning Literature Survey. Tech. rep., University of Wisconsin-Madison (2010)
49. Silva, F.L.D., Costa, A.H.R.: A Survey on Transfer Learning for Multiagent Reinforcement Learning Systems. Journal of Artificial Intelligence Research (JAIR) **69**, 645–703 (2019)
50. Silva, F.L.D., Glatt, R., Costa, A.H.R.: Simultaneously Learning and Advising in Multiagent Reinforcement Learning. In: Proceedings of the 16th International Conference on Autonomous Agents and Multiagent Systems (AAMAS), pp. 1100–1108 (2017)
51. Silva, F.L.D., Taylor, M.E., Costa, A.H.R.: Autonomously Reusing Knowledge in Multiagent Reinforcement Learning. In: Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI), pp. 5487–5493 (2018)
52. Stone, P., Kaminka, G.A., Kraus, S., Rosenschein, J.S.: Ad Hoc Autonomous Agent Teams: Collaboration without Pre-Coordination. In: Proceedings of the 24th AAAI Conference on Artificial Intelligence (AAAI), pp. 1504–1509 (2010)
53. Stone, P., Veloso, M.: Task Decomposition, Dynamic Role Assignment, and Low-bandwidth Communication for Real-time Strategic Teamwork. Artificial Intelligence **110**(2), 241 – 273 (1999). DOI https://doi.org/10.1016/S0004-3702(99)00025-9
54. Sukhbaatar, S., Szlam, A., Fergus, R.: Learning Multiagent Communication with Backpropagation. In: Conference on Neural Information Processing Systems (NIPS) (2016)
55. Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction, 1st edn. MIT Press, Cambridge, MA, USA (1998)
56. Sutton, R.S., McAllester, D.A., Singh, S.P., Mansour, Y.: Policy Gradient Methods for Reinforcement Learning with Function Approximation. In: Advances in Neural Information Processing Systems (NIPS), pp. 1057–1063 (2000)
57. Tafesse, Y.D., Wigness, M., Twigg, J.: Analysis Techniques for Displaying Robot Intent with LED Patterns. Tech. rep., US Army Research Laboratory (2018)
58. Tan, M.: Multi-agent Reinforcement Learning: Independent vs. Cooperative Agents. In: Proceedings of the 10th International Conference on Machine Learning (ICML), pp. 330–337 (1993)
59. Taylor, A., Dusparic, I., Galvan-Lopez, E., Clarke, S., Cahill, V.: Accelerating Learning in Multi-Objective Systems through Transfer Learning. In: International Joint Conference on Neural Networks (IJCNN), pp. 2298–2305 (2014). DOI 10.1109/IJCNN.2014.6889438
60. Taylor, M.E., Carboni, N., Fachantidis, A., Vlahavas, I.P., Torrey, L.: Reinforcement Learning Agents Providing Advice in Complex Video Games. Connection Science **26**(1), 45–63 (2014). DOI 10.1080/09540091.2014.885279
61. Taylor, M.E., Stone, P.: Transfer Learning for Reinforcement Learning Domains: A Survey. Journal of Machine Learning Research (JMLR) **10**, 1633–1685 (2009). DOI 10.1145/1577069.1755839

62. Taylor, M.E., Stone, P., Liu, Y.: Transfer Learning via Inter-Task Mappings for Temporal Difference Learning. Journal of Machine Learning Research (JMLR) **8**(1), 2125–2167 (2007)
63. Todorov, E., Erez, T., Tassa, Y.: Mujoco: A Physics Engine for Model-based Control. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (2012)
64. Torabi, F., Warnell, G., Stone, P.: Behavioral Cloning from Observation. In: Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI), pp. 4950–4957 (2018)
65. Torrey, L., Taylor, M.E.: Teaching on a Budget: Agents Advising Agents in Reinforcement Learning. In: Proceedings of 12th the International Conference on Autonomous Agents and Multiagent Systems (AAMAS), pp. 1053–1060 (2013)
66. Warnell, G., Waytowich, N., Lawhern, V., Stone, P.: Deep TAMER: Interactive Agent Shaping in High-dimensional State Spaces. In: AAAI Conference on Artificial Intelligence (2018)
67. Watkins, C.J., Dayan, P.: Q-Learning. Machine Learning **8**(3), 279–292 (1992)
68. Wirth, C., Akrour, R., Neumann, G., Fürnkranz, J.: A Survey of Preference-Based Reinforcement Learning Methods. Journal of Machine Learning Research **18**(136), 1–46 (2017). URL http://jmlr.org/papers/v18/16-634.html
69. Zhan, Y., Bou-Ammar, H., Taylor, M.E.: Theoretically-Grounded Policy Advice from Multiple Teachers in Reinforcement Learning Settings with Applications to Negative Transfer. In: Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI), pp. 2315–2321 (2016)
70. Zimmer, M., Viappiani, P., Weng, P.: Teacher-Student Framework: a Reinforcement Learning Approach. In: Workshop on Autonomous Robots and Multirobot Systems at AAMAS (2014)