18

Situated Learning in Autonomous Agents

Bart de Boer and Dolores Cañamero

1 Introduction

In Chapter 17 we started the discussion about autonomous agents as part of the situated cognition approach in AI, and we develop this further here, focusing on learning.

An agent is autonomous when it is able to function on its own in a complex and changing environment. We will call an agent adaptive when, in addition, it is able to improve its behaviour to make it (more) appropriate for its environment and task. Through interactions with the environment, the agent learns to improve its behaviour on the basis of its own appraisal of the perceived external or internal effects of its actions. We will thus focus on learning as a form of adaptation, where learning is an integral part of the agent's behaviour that cannot be disconnected from its performance, as opposed to a separate cognitive function that can be studied and modelled on a stand-alone basis.

In biological systems, long-term adaptation to the environment is obviously crucial for survival as is the ability to accommodate to incremental changes and to react to unexpected events. What is the significance of learning as adaptation for artificial agents? From an engineering point of view, adaptation is a design issue. When the agent is to be embedded in a dynamic, unpredictable, or insufficiently known environment, endowing it with the capability of adapting its behaviour to the context and of improving its performance over time greatly simplifies the design and construction of the system because it is not necessary to pre-wire from the outset all the knowledge the agent will be able to use.

From an epistemological point of view, autonomous agent learning can be seen as a way to cope with the grounding problem (Harnad, 1990), i.e. how the semantic interpretation of the agent's internal representations is made intrinsic to the system, rather than dependent on the interpretation that an external observer attributes to them, as happens in physical symbol systems (Newell, 1980). When representations are learned while acting in the real world, they automatically have a meaningful relation to it.

From the biological, ethological, and psychological perspectives, the study of learning in artificial autonomous agents can also act as a testbed for existing models. The analysis of learning in humans and animals can be complemented with a synthetic approach that produces or simulates learning behaviour in computer models and robots.

The major challenges encountered in autonomous agent learning stem from two main facts. First, the agent cannot stop acting in the world in order to start a learning phase when confronted with some novelty. Second, a teacher that guides or scaffolds

the agent in its learning task is seldom available in dynamic situations. At every moment, the agent will have to exploit the information it can obtain from its interactions with the environment to improve its behaviour over time and to generalize what has been learned across situations and tasks. In particular, it will have to: (a) observe the effects that its own behaviour has on/in the world; and (b) assess them as something positive, contributing to its own interests and therefore worth remembering and repeating, or as something negative and avoidable in the future. This class of learning situations and the set of techniques developed to deal with them are known as reinforcement learning in the machine learning literature (Kaelbling, Littman & Moore, 1996) due to their resemblance with this paradigm (both classical and operant conditioning) in psychology. Two main ways of applying these techniques have been used in the modelling of autonomous agent learning behaviour. The first is based on statistical analysis of the agent's observations, the other is based on Darwinian evolution of the agent behaviour.

The problems that have to be solved by these learning mechanisms are manifold. We can consider them as caused by two factors: the need for the agent to survive and the complexity of the (interactions with the) world. The drive for survival prevents the agent from spending all its time exploring the world — it also has to efficiently exploit what it already knows about the world. A mechanism is thus needed to continuously arbitrate between exploration and exploitation (Wilson, 1996). Since the agent cannot stop the world, learning must take place while it simultaneously works on survival. Furthermore, the learning mechanism should be flexible enough to allow the agent to react to sudden changes in its environment. A learning mechanism that can learn one task very well, but that is not able to unlearn it, is not sufficient. Also, the agent needs to be able to distinguish between what is useful and what is not useful to learn.

The environment in which artificial autonomous agents usually operate — partially observable, noisy, real-time, changing — also poses a number of problems. If the agent performs an action usually the reward (or punishment) arrives after a certain time. This reward then has to be correctly assigned to the right action — the credit assignment problem. This becomes more complex when a number of different actions have been performed either sequentially or simultaneously. Since the agent's perception of the world can be extremely inconsistent, and as the agent cannot stop performing in order to learn, forming a useful world model becomes a difficult task.

Two types of situated agents, synthetic and robotic, are now discussed, as there are a number of interesting problems and possibilities particular to these two different embodiments.

2 Learning in Software Implementations

Software implementations of autonomous agents avoid some of the specific problems posed by physical (hardware) embodiment as well as those arising from having to deal with the physical world (see next section). For this reason, they usually offer more freedom to experiment with novel architectures and learning techniques. In particular, they allow for an easier integration of symbolic and nonsymbolic representations and learning models. In this section, we will examine three landmark architectures that can be considered hybrid in the sense that they attempt to combine the best of both worlds: classifier-system-based animats, behaviour networks, and a schema-based simulation of the Piagetian development theory. Although the three integrate a (more or less structured) symbolic representation within a neural network computational model, each of them illustrates a different type of learning technique. They are all instances of synthetic agents — autonomous agents that inhabit computer-simulated environments — or, in other words, simulated robots inhabiting artificial worlds.

2.1 Animats

Animats are simple artificial animals created with the aim of studying intelligent behaviour at a primitive level where, "... intelligent behaviour is to be repeatedly successful in satisfying one's psychological needs in diverse, observably different, situations on the basis of past experience" (Wilson, 1985). They are adaptive systems designed to operate in a closed-loop interaction with their environment, from which they receive signals in the form of binary feature vectors and on which they can perform a predefined set of actions that change the signals. Only part of these input signals are relevant at each moment, and some of them (or their absence) have a special status for the animat, being perceived as a reward. Past experience is taken into account by keeping an internal record of earlier interactions with the environment and their results.

The basic problem of animats is to generate rules that associate sensory signals with adequate actions so as to maximize the amount of reward, and therefore survive and move around more efficiently in their world. This problem is decomposed in three parts: (1) to come up with efficient rules; (2) to discard inefficient ones; and (3) to generalize the rules that are kept in an optimal way. Different animats have been implemented using classifier systems (Holland, 1992; Holland & Reitman, 1978). These are parallel message-posting systems that learn binary condition–action rules, called classifiers, which compete to control the behaviour of the system. Learning classifier systems are composed of three main elements: a performance module, an apportionment of credit system, and a genetic algorithm. The performance module is responsible for action selection. It consists of a set of finite-length classifiers and a finite-length message list of binary strings. Classifiers are defined over a binary alphabet plus a wildcard or 'don't-care' symbol that allows for some generalization. At each time step, incoming environmental messages — perceptions — are posted to the message list, where the messages activate those classifiers whose condition part matches the message; these classifiers may in turn activate others or may cause an action to be taken by the animat. Conflict among competing classifiers is solved by the outcome of an activation auction, which depends on the evaluation of the weight (usefulness) of each classifier. This value — the classifier's strength — is assigned by the apportionment of credit module according to each classifier's role in obtaining reward from the environment. This module is thus a form of reinforcement learning

that stores stimulus–response (S–R) associations in their simplest form, i.e. for purely reactive systems, or S–R–S associations for systems that learn more complex models of the world that predict not only rewards, but also world states, allowing therefore, for some anticipation, such as in Riolo (1991).

At a certain number of time steps, the genetic algorithm introduces new, better rules into the system by applying a tripartite process of reproduction, crossover, and mutation to the existing ones. Rule strength is used to select the best classifiers that will generate the new ones. The genetic algorithm usually comes up with more general classifiers — those containing more wildcards — that allow the system to use fewer rules, possibly organized in default hierarchies, to model more complex worlds with (relatively) many states. However, transfer across tasks remains difficult, since once a rule has been assigned a given strength for having solved a task, current classifiers are not able to use it for a different task, perhaps due to the fact that their condition part is not sensitive to the animat's goals or motivations. Also, the choice between exploitation, done by the performance module, and exploration, carried out by the genetic algorithm, is done in an ad hoc manner, since it is the system designer, based on experimentation over many trials, who decides how often the genetic algorithm comes into play.

2.2 Behaviour Networks

This architecture was proposed by Maes to model action selection as an emergent property of a parallel process in such a way that the system can show various trade-offs between reactive and more deliberative behaviour.

In this approach, an autonomous agent, a creature, consists of sets of fixed behaviours in which it can engage, which constitute the nodes of the network. For example, the creatures presented in Maes (1991) inhabit a two-dimensional computer-simulated world containing obstacles, food and water sources which they sense at very short time intervals, and in which they can perform behaviours such as feeding, drinking, sleeping, exploring, approaching or fleeing from other creatures, etc. They also have motivations to act: safety, curiosity, fear, hunger, etc. Behaviours are implemented using STRIPS-like operators which, in addition to preconditions (stimuli), add- and delete-lists, consist of a set of processes implementing the behaviour, an activation level that reflects their relevance for the perceived situation and an activation threshold that must be surpassed for the behaviour to become active. Behaviours are linked through three types of causal relations — successor, predecessor, and conflictor links — defined according to the relations between propositions in their precondition, add- and delete-lists.

Behaviour selection is the emergent functionality of a homeostatic process involving the behaviours and motivations (goals) of the creature and the state of the environment. This process spreads both activation and inhibition between behaviours using the above links so that the activation energy accumulates in the behaviours that represent better action choices for the current situation.

In Maes (1992) weights or probability estimates of the reliability of causal relations between the behaviours were added in order to exploit environmental feedback which improves the network as the creature accumulates more experience about the actual effects of its behaviours in the world. Weights are used in both the action selection and learning mechanisms, performance and learning being fully integrated. The learning task is, for every behaviour, to learn its successor, predecessor and conflictor links together with their respective reliability, when the creature is acting in a noisy, dynamic and non-deterministic environment. This learning algorithm presents many interesting features. The probabilistic behaviour selection mechanism leads to an elegant trade-off between exploration and exploitation, since the system reduces the amount of experimentation as learning progresses and the links take values approaching 0 and 1. It also uses a priori knowledge that allows the designer to test how different factors bias learning by controlling the degree to which the creature's knowledge is innate versus learned. It also allows for some transfer since the links learned in the context of one motivation can be applied in the context of another. As for its relation with human (and animal) learning, this algorithm partially models both instinctive learning and instrumental conditioning. In return, the algorithm makes a set of simplifying assumptions that limit its applicability, although some possible extensions are sketched in Maes (1992). For example, it assumes that the effects of behaviours are very immediate, a behaviour being completed before the next one is activated, which is seldom the case in many real applications, especially involving robots. The creatures are supposed to know what behaviours achieve the different motivations.

Finally, the repertoire of innate behaviours is assumed to be sufficient and granular enough to allow the creature to learn how to achieve the motivations, and therefore it does not support the learning of new behaviours. This latter type of learning is developed in the architecture we examine next.

The Schema Mechanism

This general learning and concept-building mechanism constitutes an explicit attempt to replicate key aspects of human cognitive development with AI techniques, taking Piaget's theory as a source of inspiration (Drescher, 1991). Following Piaget's constructivism, the mechanism uses almost no a priori knowledge of the world. As a consequence, the system should be able to learn in a broad range of environments and transfer its knowledge from one to another more easily than a strongly biased one; it should also have powerful exploration capabilities, as it will have to make discoveries far beyond the knowledge it has built in. In return, such a system faces two main problems that can make learning very slow: interpreting its experiences in a non-deterministic world without the guidance of a bias in such a way that it can learn from them, and inventing new concepts suited to the different problems it faces. Drescher (1991) implements this mechanism in a simulated robot embedded in a discrete bidimensional microworld that very roughly replicates the world as perceived by an infant younger than 8 months. The cells of the world can be either empty or

occupied by one object. The robot consists of a one-cell body/head that cannot move and a one-cell mobile hand disconnected from the body. It also has a mobile eye with a square visual field, the centre of which — the fovea — can see additional details in objects so as to determine their identity. The mechanism uses three kinds of data structures: items, actions, and schemas. Items convey sensory information — either proprioceptive, tactile or gustatory — in the form of propositions about the state of the world, and can be on, off, or in an unknown state. Actions designate events that can affect the state of the world. Schemas are complex units of knowledge, both declarative and procedural, reflecting regularities. A schema comprises a context, an action, and a result, stating that if the action is performed when all the context conditions are satisfied, the result conditions will hold (i.e. these items will be turned on) with a certain degree of reliability that the schema maintains. The system is initially provided with some primitive sensorimotor data structures of every category. However, this built-in knowledge is encapsulated to each sensory system and is not available to the schema mechanism, primitive structures being featureless entities for it. Based on its experiences while randomly interacting with the world, the system learns: new items to designate novel aspects of the world; new schemas that connect items and actions to express discoveries about regularities in the world; and composite actions that achieve a result independently of which schemas are used for that, therefore allowing it to represent events at levels of abstraction higher than the sensorimotor one, and to designate externally caused state transitions.

Some of the typical learning achievements of the system are the elaboration of intramodal schemes about body-related positions adjacency; schemas for intermodal coordination such as visual effects of hand motions that allow the system, for instance, to anticipate tactile contact when the hand is seen in motion beside an object, or the elaboration of synthetic items that begin to designate objects as distinct from their perceptions (object persistence). This system presents two main drawbacks. First, it is not coupled to a performance system, and therefore the closed-loop interaction with the world is missing. Second, learning is too slow — it is not even guaranteed to happen — because there is nothing to guide the learning, neither *a priori* knowledge nor internal motivations. Foner (1994) has attempted to overcome this second problem by integrating an attentional system into the schema mechanism that makes learning more selective by having the system focus its attention on what is important for it.

3 Learning in Robotic Agents

The problems of autonomous learning are amplified by making a physical embodiment, or, in other words, a robot implementation of the learning system. Not only does the robot have to cope with the general problems of autonomous learning, such as the lack of a teacher, the assignment of credit and the problem of performing and learning at the same time but it also has to process complex sensor data, plan complex actuator actions and operate in real time (Smithers, 1996).

Let us first make clear what kind of robots we are talking about. The robots that are used are of course not at all like the humanoid robots known from science fiction, nor are they like the robot manipulators found in car factories and the like. They are usually small vehicles, between 10 cm and 1 m in size, usually with three or four wheels, sometimes with simple legs and equipped with a number of sensors, such as a camera, touch sensors, light sensors or infrared sensors. Generally they have a small, but powerful, computer on board, together with a rechargeable battery, so that they can operate completely autonomously for a reasonable amount of time.

At the moment a number of projects are underway that try to build more human-like robots, with complex sensors and manipulators. Note that in the design of these robots a number of important trade-offs must be made. If one equips the robot with limited sensing abilities, it will be able to process more sensor data per unit time but it will have a less accurate view of its environment. If it has more powerful sensors, it will need more time to process the sensor data. If one makes a big robot, one can equip it with big batteries, a big computer and lots of sensors, but it will also be more difficult to do quick experiments or to build more robots in order to do experiments with multiple robots. If one builds a small robot, it is easy to do quick experiments, and easy to build many of them, but the computing and sensing abilities will be less. Different laboratories have made different design decisions (Donnet & Smithers, 1991; Mondada, Franzi & Ienne, 1994; Brooks & Stein, 1994).

The learning paradigms that are implemented on autonomous robots are the same as the ones in autonomous synthetic agents. There is on-line learning, in which individual robots try to improve their behaviour over time and there is evolutionary learning, in which a population of robots tries to improve its behaviour over time. Both are trying to solve the problem of assigning the reinforcement they get to the right behaviours. The first (Gaussier & Zrehen, 1994; Lambrinos, Scheier & Pfeifer, 1995; Mahadevan & Connell, 1992; Marjanovic, Scassellati & Williamson, 1996) does this by estimating which actions of a robot were responsible for the reinforcement and by making it more likely that these actions will be performed in a future similar situation. The second (Cliff, Harvey & Husbands, 1993) assigns reinforcement by looking at a large number of robot control systems, considered good if they cause a robot to survive long enough. A new generation of robots is then equipped with small variations of the best control systems. These control systems are tested in the same way and the cycle is repeated.

The fact that a robot has to operate in real time places time constraints on the learning mechanism but this can usually be overcome by using a more powerful on-board computer. The real-time requirement is much more of a problem for the sensing of the environment because in a limited amount of time only a very limited amount of knowledge about the environment can be acquired.

The problems with sensors and actuators are often underestimated in learning robot research or are considered not to be central to the issue of learning. In most experiments, learning tasks and environments for robots are constructed to provide just the right sensor cues for the learning system so it is not necessary to worry about the sensor processing. For practical applications, however, and in order to achieve results that are applicable to learning in humans and animals, one would like to have a learning robot that is able to extract the necessary information in a realistic environment instead. In fact, in order to understand learning in humans and animals and in order to be able to build learning robots for realistic applications, an understanding of sensor processing and actuator planning seems essential. No known learning system can cope with a continuous stream of data so it has to partition the continuous stream of sensor data into events, both in time as well as in space. Examples of such events can be the detection of a charging station, the touching of a wall, or the performance of some useful task. These events can then be associated with actions and with increasing or decreasing fitness. The partitioning of continuous data streams into events is very hard and rather task-specific.

A related problem is that of sensor fusion. Sensor fusion is the use of different sensors in combination in order to be able to distinguish certain events better than the individual sensors could. Robot control systems that can fuse sensors autonomously operate better than ones that cannot. The robot must also calculate its own reinforcement from what it perceives through its sensors, as there is no external teacher to provide it with the reinforcement signal. Often, given a certain learning task, it is not directly obvious how the reinforcement should be calculated and sometimes it is necessary to have different reinforcements for the different sub-behaviours of the robot.

Furthermore, new actuator actions should be made up. This is also no trivial task as, given a certain goal the agent wants to achieve, it is not always easy to calculate the exact actions of the robot's motors that have to be performed. Usually some kind of feedback mechanism is necessary where a sensor is used to monitor and guide the movements of the actuator. A robot arm can serve as an example here: given the task of grasping a certain object at a certain location, how should the motors be activated? If a camera to sense the position of the robot arm is used it is easier to guide it to its destination.

Another difficulty is that of the inherent unreliability of real-world sensors and actuators. A sensor never gives exactly the same signal to the same stimulus, nor does an actuator always give the same response to the same command. Sensors and actuators also change over time: motors get hot, sensors get displaced, actuators wear, and usually performance is dependent on the amount of energy in the robot's batteries. Of course, one could try to compensate for this by using better sensors, actuators and batteries but, in reality, it appears to be necessary to tune the sensors and actuators constantly.

These problems are generally tackled by designing task-specific subsystems for the different levels on which learning is necessary. These are meant to function in the environment for which the robot is designed. The subsystems are for partitioning the sensor-streams into events, for inventing and planning new actuator actions, for fusing sensors, for calculating the reinforcement and for tuning the sensors and actuators. The actual interesting learning is performed by the same types of general learning mechanisms that can be found in autonomous synthetic agents. However, because the inputs of these learning systems depend on the many environment-specific subsystems that process sensor data, the overall learning control system of the robot is environment-specific as well.

The problems can be illustrated by an example of a simple robot learning experiment which has been performed at the AI laboratory of the Vrije Universiteit Brussel. The robot is equipped with two wheels and two motors and steers by running one motor faster than the other (differential steering). Its sensors are touch sensors and infrared sensors. It has been programmed to retract if its touch sensors are touched, thus effectively avoiding obstacles. The robot's performance would be better if it could use its infrared sensors to avoid obstacles since it would then be able to avoid physical contact with them. However, it is rather hard to predict what kind of signals the infrared sensors will produce if the robot gets near an obstacle, so learning (or rather, self-tuning) is necessary.

Although this task is rather trivial, it illustrates all the problems of learning in robots. It has to happen on-line (while the robot is running) and it has to be self-reinforcing (there is no external teacher). The robot has to filter events from the stream of sensor inputs because it should only learn when it is near an obstacle. Furthermore the explore/exploit problem has to be tackled: in order to learn, the robot needs to encounter obstacles but, in fact, the learning goal is to avoid them. Also, the task is an example of sensor fusion: infrared and touch sensors are used in co-operation to avoid obstacles.

This particular problem was solved by designing an ad hoc learning mechanism. The performance part consisted of connections with a certain strength between the infrared sensors and the motors. If the infrared sensors gave a strong signal, and if the connection with the motor was strong, the motor would run fast. If there was little signal, the motor would run slowly. The connections were initialized with a very low strength. But whenever an infrared sensor and a motor were highly active at the same time and a touch sensor had just been touched, the connection between that motor and that infrared sensor would be strengthened. This learning mechanism is called Hebbian learning (Hebb, 1988). This algorithm solves the learning task. Every time the robot has just hit an obstacle, the motors are running in such a way as to avoid the obstacle. The infrared sensors are close to the obstacle so they give a strong signal. After a while the robot will start making the same steering actions through the signals of the infrared sensors as when the touch sensors were touched.

One problem is that of overtraining: if the connections become too strong, the robot will stay too far away from obstacles which would prevent it from navigating efficiently. Therefore the strength of the connections has to decay slowly. One could say, with the risk of taking a human metaphor too far, that the robot forgets how to avoid obstacles. This also solves the explore/exploit problem. If the robot bumps into obstacles very often, its connections are strengthened until it does not bump into obstacles any more. But because the strength decays slowly, every once in a while it will bump into an obstacle again, so it makes sure it does not stay too far away from the obstacles.

This particular learning mechanism is completely situated. It only works because in the robot's environment, infrared signals and touch sensor signals correlate. It can only touch an obstacle if it is close by and then the infrared sensors will also give a strong signal. Also the explore/exploit problem is solved through the environment; connection strengths decay until the robot bumps into an obstacle again. If the obstacles were to be removed, the strengths would decay to zero.

Learning in autonomous robots crucially depends on the ability to partition a robot's sensor streams into meaningful events, and to derive information on how well it is performing from its sensor inputs. Humans and animals are very good at this, and research into human and animal learning often assumes this ability as a given. Research into autonomous software agent learning can avoid these difficulties because, with synthetic agents, the environment is usually designed to deliver exactly the right data for the software agent to perform (although one tries to simulate more and more complex environments). The research into robot learning has pointed out that handling the stream of sensor inputs is no trivial task, and that a lot of the apparent intelligence of a robot resides in its ability to process sensor data. As the sensor data which the robot receives is completely dependent on its environment the learning such a robot can do is inherently situated. Indeed, it would be impossible to build a learning robot (by hand or through evolution) that can operate independently of its environment.

4 The Role of Situatedness in Learning as Adaptation

From the machine-learning perspective, what does the situatedness aspect contribute to learning, as opposed to disembedded symbolic approaches? From constructing artificial autonomous learning systems we have learned that situatedness can present both advantages and problems. From the engineering viewpoint, perhaps the most obvious advantage comes from the fact that having an agent learn from its experience and adapt to the environment can dramatically reduce the design element. Moreover, a situated learning system avoids being faced with the grounding problem, and is more truly autonomous. Situatedness also allows for efficient learning — indeed, the most efficient and effective learning and behaviour are often very situation-specific because the learning algorithm for a given environment is specialized (either by design or through self-adaptation), in the same way as a behaviour is specialized. This means that a certain learning mechanism can be highly adapted and useful in a given environment, while in a different environment it can be almost useless. The negative side is that transfer becomes a hard problem, and it would seem that situatedness and transfer are in contradiction with each other — the more situated the system, the more difficult transfer is.

Transfer can be seen at several levels in autonomous agent learning. The first is transferring the agent across environments: as with biological agents, this is usually possible — the agent adapts to the new situation — as long as the new environment is not significantly different from the previous one, i.e. it is the same type of ecological niche. The second is transfer of learned behaviours across tasks. This type of transfer is not easy for most reinforcement learning algorithms, in particular when (a) the goals or motivations that guide the agent's behaviour are implicit, and (b) what is learned is just S–R associations based on sparse reinforcement signals coming from the environment. Once an association between a (set of) behaviour(s) and a goal or task is learned it is very difficult to associate those same behaviours to a different task. One typical solution is to have the system forget by using a decay function that progressively decreases the strength of the association so that it can relearn a new one (or the same) when needed. However, in this case we cannot say that the agent is transferring previously acquired knowledge, but rather re-adapting to a new situation.

Apart from the difficulties mentioned above, there are two other partial reasons for the difficulty these algorithms have in dealing with transfer: their very limited use of internal states and the absence of variables in the traditional sense which makes it hard for the agent to keep a good memory of its actions. In particular, we are dealing with distributed and non-linear systems whose global behaviour is an emergent property of local interactions and, therefore, it is usually not possible to single out which elements and structures are responsible for a certain behaviour and can be usefully reused.

The complexity of the environment, usually very dynamic, uncertain and noisy in non-trivial learning situations, highly increases the complexity of the learning task. This poses problems concerning the processing of sensory data and the division of the sensory stream into meaningful events, in particular for robotic agents. However, although these problems complicate the learning process, they affect the sensory processing and motor contol aspects more than the learning mechanism itself. It is a non-negligible problem, though, especially if we take into account that the environments and tasks that have been used for autonomous agent learning are relatively simple, and therefore scaling up becomes a very hard endeavour.

From the perspective of the designer, transferring the experience acquired from building one system to another is also problematic since the results of research on autonomous agent learning are often rather hard to interpret and evaluate. This is in part due to the task-driven, pragmatic approach adopted to design learning systems and the experiments created to test them.

Until now, the field has been more concerned with building working (and demonstrable) agents than with following a rigorous scientific approach, which is still a long way from the current state-of-the-art. For example, not enough is known about the different environments to come up with a good classification of them, although some attempts have already been made in this direction (Littman, 1993; Todd & Wilson, 1993). As a consequence, there are no standard environments in which to perform experiments that would allow better comparisons of results. The same is true for the learning tasks. As far as learning performance is concerned, neither do we have good standards, nor are there standard means (or sets of criteria) to compare and evaluate the performance of agents. Even more importantly, we have been lacking good theories and models about fundamental concepts and issues, and only now do we start to see the first steps: the relationships between learning and other not fully understood problems such as action selection, adaptation, evolution, development, or emergent functionality. Also, researchers in this field have taken for granted a simplified version of only one learning paradigm — behaviourism — disregarding other models that only now start to be acknowledged as relevant. As Maes puts it, ". . . the agents built using this approach end up looking more like 'a bag of hacks and tricks' than an

embodiment of a set of general laws and principles. Does this mean that the field will evolve into a (systems) engineering discipline, or will we find a path towards becoming a more scientific discipline?" (Maes, 1994). We are not in a position to answer this question yet, so we let the reader guess where our hopes and efforts are directed.

5 Conclusion

In this paper we have examined and illustrated how AI approaches the problem of learning in artificial autonomous agents. Situatedness is seen to be essential to this kind of learning, although it also poses a whole range of problems which are very difficult to solve given the current state of the art. Even though the results in the field cannot be considered spectacular, we believe that the study of learning as adaptation in situated agents is the right approach for achieving meaningful learning in artificial systems.

At this point, one might ask what is the significance of our research for research in human learning. We see several important contributions. The study of learning in artificial autonomous agents offers a synthetic approach to learning and intelligence that can nicely complement the work on human (and animal) learning, which traditionally adopts an analytic perspective, in the sense that it is concerned with the study of existing systems. In this respect, the study of learning in autonomous situated agents is more relevant to human and animal learning than the study of non-situated learning since humans and animals are also agents that must continuously operate in their environment.

The operationalization of psychological theories in autonomous agents can also help to improve understanding of these theories, to test and analyse their predictions, and to contribute to their usefulness or validity in a direct way. Some interesting observations have already been made in this direction concerning, for example, the tight coupling between learning and performance, the development and use of dedicated and environment-specific learning subsystems, or the cruciality of knowledge about the task and the environment for some inference processes. Some projects, such as the humanoid robot Cog (Brooks & Stein, 1994), explicitly advocate the use of ideas from developmental psychology in their design. This has attracted the interest of psychologists and we begin to see studies about the parallelisms between infant development and the behaviour-based approach to AI and learning and its implications for the psychological study of human development (Rutkowska, 1994). Many important technical and methodological problems in autonomous agent learning have yet to be solved. Among the latter, perhaps the most important ones stem from the ad hoc manner in which we have approached our designs and learning algorithms. We believe that a close collaboration with researchers from other disciplines, such as neurosciences and psychology, can be the first step to take in order to overcome them.

Acknowledgements

We are indebted to Joan Bliss for her very valuable suggestions and her continuous support, and to the VUB AI-Lab for providing an excellent research environment. Bart de Boer was financed in part by the Belgian federal government project on emergent functionality (FKFO contract no. G.0014.95) and the IUAP Construct project (no. 20). While writing this chapter, Dolores Cañamero was a postdoctoral fellow at the MIT Artificial Intelligence Laboratory. She wishes to thank Rodney Brooks and the Zoo group for their support, and for very valuable discussions on this and related topics. Her research was funded by the Spanish Ministry of Education under grant PF9500410164.