A survey on Machine Learning in Mobile Robotics

João Cunha

Abstract – **Learning is a prerequisite for intelligent behaviour. It is no surprise that Machine Learning, as one the most important fields of Artificial Intelligence, is becoming an increasingly hot-topic in Robotics. Machine Learning has several benefits, from enabling the use of historical data to improve future decisions, to solving problems that are difficult by hand-coding the solutions or to allow adaptive behaviours in highly dynamic environments such as learning individual preferences in human-computer interaction environments. Recent advances in algorithms and the increasing computational power available at reduced size and weight, enabled the application of Machine Learning to the field of mobile robotics. This report presents an overview of the state of the art of the different applications of Machine Learning methodologies in mobile robotics.**

I. INTRODUCTION

Given the high degree of multidisciplinarity in the field of Robotics, programming a robot poses several challenges. Firstly there are no high-level programming languages to aid the development of robust algorithms. Secondly the robot sensors and actuators are complex to model. Thirdly robots have been steadily moving towards more unstructured environments where robot programming becomes a arduous task.

The aforementioned are just a small set of the reasons why researchers have been working on enabling robots to learn how to perform tasks by themselves. Thus the area of Robot Learning is the application of Machine Learning methodologies to the field of Robotics.

According to [1] the definition of Machine Learning is: "Any computer program that improves its performance at some task or class of tasks through experience."

Therefore every Machine Learning problem is characterized by:

- a task or a class of tasks (ex: playing chess);
- a performance measure (ex:percentage of games won);
- experience (ex:playing practice games).

There have been several different machine learning methodologies developed over the years, however all are characterized in terms of supervised and unsupervised learning. These two classes differ in the sense that in the former there is a "teacher" supplying the learning program with training data, while the latter has significantly reduced or nonexistent feedback on the learning task.

Although different Machine Learning methodologies have been successfully applied at solving complex and non-trivial problems in the field of Robotics, such as estimation of sensor noise [2], environment representations [3] [4], or control policies [5], making robots learn is still an open challenge. Factors as sensors and actuators noise impose difficulties in learning problems. Additionally [1] describes learning as a search problem of finding a policy that best fits the training examples, hence high dimensional or continuous state spaces are prohibitive factors in learning problems. Overfitting the training data is also a concern in Machine Learning since the learning program can demonstrate very high performance in the presence of training examples while failing to generalize a policy for future examples.

The remainder of this paper is structured as follows. Section II presents the Credit Assignment Problem the basic problem of any learning problem. Section III describes some of the most important learning paradigms applied to the field of Robotics, along with some illustrative examples. Finally section IV presents the conclusions.

II. THE CREDIT ASSIGNMENT PROBLEM

Robot Learning is a problem of learning a policy π from a set of sensory states S to a set of responses R.

Learning a policy requires solving three credit assignment problems [6]. The temporal credit assignment involves giving credit or blame to a given response, in a sequence of responses, for a good or bad outcome. The structural credit assignment determines the range of sensor values that yield the same outcome. Finally the task credit assignment generalizes a sequence of responses to perform other similar tasks.

The various learning paradigms, presented in the next section, are characterized by solving each credit assignment problem in a different manner.

III. LEARNING PARADIGMS

This section presents four different Machine Learning paradigms that have been successfully applied in field of Robotics to solve various tasks.

A. Inductive Concept Learning

The first paradigm presented is Inductive Concept Learning. In this paradigm there is a teacher providing training data along with the classification of the same data. Therefore this is a supervised paradigm where the temporal credit assignment problem is solved by the teacher. The learning problem is then reduced to

inferring which values from the attributes of the training data actually affect the value of the classification.

Inductive Concept Learning is a well known paradigm not only in Robotics but in Machine Learning in general. Hence various different methodologies have been developed. Of the most important methods Version Spaces, Decision Trees and Neural Networks are of notice. In particular Decision Trees and Neural Networks have been successfully applied in Robotics for their ability to cope with uncertainty and noisy data.

An additional concept very important in learning in general and in Inductive Concept Learning in particular is the ability to generalize for unseen data. This is known as inductive bias [7], in the sense that the learner is provided *a-priori* assumptions on the target policy. An example of inductive bias used in Machine Learning is Occam's Razor.

A.1 Examples

There are a variety of examples of the application of Inductive Concept Learning in the field of Robotics, some dating more than 20 years. In fact as early as 1988, Pomerleau [8] was able to teach an autonomous car to drive in two-lane highways by training a neural network from previously captured images of a driven car. Such Learning from Demonstration (LfD) method was applied in the 2005 DARPA Grand Challenge winner Stanford car Stanley to perform highway [9] and car lot [10] navigation. A different application of inductive concept learning is present in [11] where the Sony AIBO platform was able to determine entangled or stuck status and even the type of surface it was walking on based on accelerometer data.

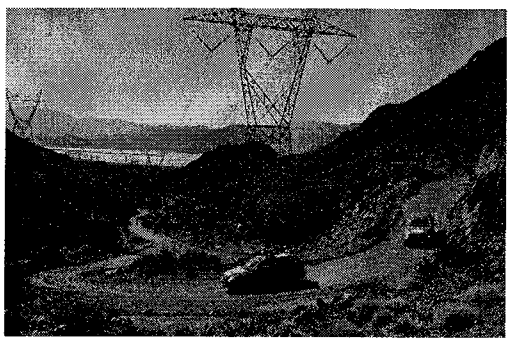

Fig. 1 - The Stanford car Stanley, adapted from [12]

B. Explanation Based Learning

Explanation Based Learning is another supervised learning paradigm. However, in Explanation Based Learning the teacher doesn't provide a classification along with the data. Instead provides a domain theory of how the training examples are consistent with the target policy. Thus not only the temporal credit assignment but also the structural credit assignment problems are solved by the teacher.

The domain theory can be provided in various forms, from logic rules to neural networks. The domain theory can be viewed as an *a priori* knowledge of the task to be perform, and thus allows to speedup the search over the space of possible hypotheses.

B.1 Examples

Explanation Based Learning was a hot-topic in 1990 decade, where computational power was scarce and the domain theory was highly regarded as it enabled to speedup the learning process when compared to other paradigms at the time. However, the increased requirement is currently a derogatory factor when opposed to other learning algorithms which require less *a priori* requirements.

Nonetheless, [13] presents a remarkable example of an application Explanation Based Learning. In this case the domain theory is a neural network modelling each action of the robot. Reported results show that in 10 minutes the robot learned to navigate towards a landmark in an office environment.

C. Evolutionary Learning

Evolutionary Learning is a very distinct learning paradigm since it is not inspired on human reasoning but is a close analogy of biological evolution [14].

In Evolutionary Learning, the learner is only provided with a fitness function and a target measure. Hence this paradigm is considered a unsupervised learning paradigm. The learner searches for the optimal policy from an initial random set of hypotheses. According to the provided fitness function, the most fit individuals of the population are chosen to generate the following generation. While some members pass intact to the following generation (reproduction) others are combined with each other to produce new offspring (crossover). Aside from the genetic operators of reproduction and crossover, some elements of the population maybe suffer changes resulting in different individuals (mutation).

Thus Evolutionary Learning searches for the optimal policy through search space of possible hypotheses by generating variants of the best current hypotheses.

There are two major variants of Evolutionary Learning: Genetic Algorithms and Genetic Programming. The basic difference is that in Genetic Algorithms hypotheses are encoded in strings while in Genetic Programming the hypotheses are encoded in computer programs.

C.1 Examples

Examples of Evolutionary Learning in the field of Robotics can be found in gait learning such as in [15] where the gait of a biped simulated humanoid was obtained using Genetic Algorithms.

Also of notice is one of the driving forces of Evolutionary Robotics, Hod Lipson, who focuses on applying Evolutionary Learning not only to robot controllers (brain) but has also applied Evolutionary Learning to the robot hardware (body) achieving evolved morphologies from initial simple robots [16].

D. Reinforcement Learning

The last learning paradigm presented in this paper is Reinforcement Learning [17].

The basic framework of Reinforcement Learning is a Markov Decision Process (MDP). MDP are characterized by a set of states S, a set of actions A, a state transition function $\delta : S \times A \rightarrow S$ and an immediate reward function $r : S \times A \rightarrow \mathbb{R}$. A fundamental characteristic of MDPs is that a state s only depends on a finite number of past states.

The learning problem is to find a policy $\pi : S \rightarrow A$ which produces the greatest cumulative reward over time, $u(s)$. The greatest cumulative reward gained in a given state is given by $u(s) = \sum_{i=0}^{\infty} \gamma^i r$, where γ is a discount factor of the delayed rewards in the future.

Reinforcement Learning methodologies are impacted by factors such as delayed rewards, since the robot might only receive a positive reward when it reaches the goal state, what may take some time to achieve. On the other hand while learning the robot must choose between exploiting a previously learned policy or to explore unknown states and actions. Finally, the robot sensors may not be enough to observe the entire surrounding environment.

An optimal policy is then a policy that for a given state s chooses the action a that maximizes the immediate reward of applying a in state s plus the cumulative reward of the successor state $\delta(s,a)$. This policy is given by $\pi(s) = \text{argmax}_a [r(s,a) + \gamma u(\delta(s,a))]$.

Hence a robot with the perfect knowledge of the state transition and the immediate reward function can determine the optimal policy by applying the value-iteration algorithm which is proved to converge to the optimal value.

Here we assumed a deterministic state transition function. However the state transition function is commonly probabilistic given the sensors and effectors errors and noise. Thus the optimal policy can be extended to accommodate probabilistic state transitions, $\pi(s) = \text{argmax}_a \left[E[r(s,a)] + \gamma \sum_{s' \in \delta(s,a)} P(s'|s,a) u(s')\right]$.

However having a complete and perfect knowledge of the state transition function is often an unrealistic scenario, compared to having a perfect domain theory in explanation based learning.

To overcome this limitation a model-free Reinforcement Learning methodology is usually used, Q Learning. Q Learning is based on learning the Q function, $Q(s,a)$ which represents the maximum discounted cumulative reward obtained from state s and applying action a [1], $Q(s,a) = \left[E[r(s,a)] + \gamma \sum_{s' \in \delta(s,a)} P(s'|s,a) u(s')\right]$. On the other hand $u(s) = max_{a'} Q(s,a')$ which rewriting the previous equation yields $Q(s,a) = \left[E[r(s,a)] + \gamma \sum_{s' \in \delta(s,a)} P(s'|s,a) Q(s',a')\right]$.

To learn the value of the Q function the robot starts at a random initial state s and applies an action a while observing the resulting state s' and the obtained reward r. Hence Q-Learning can be viewed as a robot acting randomly upon the environment and analysing the outcome of its actions. This is usually done using a table with an entry for each state-action pair. This is major constraint since the robot estimation of the Q function will only converge is every pair state-action is visited sufficient times. This is an unrealistic assumption for very large dimensional or continuous spaces. On the other hand the learned policies would not be capable of generalizing to unseen examples. An alternative approach is to use neural-networks instead of explicit tables, in what is known as neural Q function. While this alternative has advantages and has been successfully applied in various robotic systems, classic algorithms for training neural networks to accommodate a new pair state-action, may change the Q estimates for other state-action pairs. This fact affects the convergence towards the optimal value and explains the large learning times and several thousand experiences needed to achieve a good policy.

However variant algorithms were proposed in order to minimize the number of training examples required. Of notice is the method Neural Fitted Q Iteration [18] which stores the previous training examples as tuples s,a,s',r that are considered when the neural network is updated for new experience data. This method is reported to be able to achieve a close to optimal policy in just a few hundreds examples.

Reinforcement Learning is very important in robotics as a framework for autonomous learning given the very few requirements when compared to other paradigms. Thus Reinforcement Learning is an unsupervised learning paradigm.

D.1 Examples

As mentioned before Reinforcement Learning algorithms have been widely used in the field of Robotics with great success. One special application of Reinforcement Learning is robot control.

One example of such application is a classical control problem, the inverted pendulum problem. Riedmiller [5] was able to balance a single and double inverted pendulum using only Reinforcement Learning.

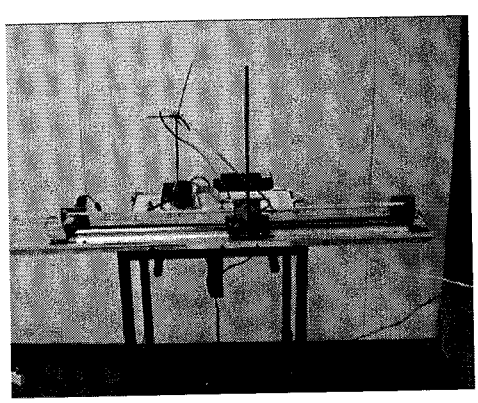

Fig. 2 - The inverted pendulum system used in [5].

A prolific environment for Reinforcement applications is the RoboCup robotic soccer competitions which have numerous examples of Reinforcement Learning. For instance tasks from low-level control, such as motor control, behaviours, such as ball interception and dribbling, to cooperation skills, such as attacking strategy have all been solved using Reinforcement Learning [19] [5] [20].

Another example of Reinforcement Learning in particular in the RoboCup competitions can be seen in the RoboCup Multi-Agent Special Interest Group site [21] where it is shown that the vast majority of the learning methods applied is Reinforcement Learning. In particular the now extinct four legged league (4LL) is a great example of the application of Reinforcement Learning methods to complex-modeled systems such as the Sony AIBO platform.

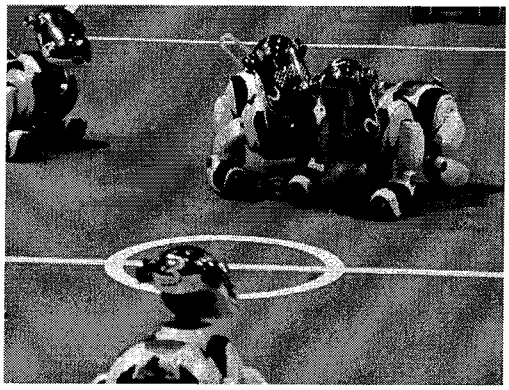

Fig. 3 - A game of the RoboCup Four Legged League, adapted from [22].

IV. CONCLUSIONS

This paper presented an overview of Machine Learning applications in the Field of Robotics. Robot learning offers several advantages for developing robotic systems while replacing hand-coded methodologies which assume the programmer has a perfect knowledge of the system model in order to produce optimal algorithms. This paper presented the different types of credit assignment problems that must be solved in Machine Learning. Finally four different learning paradigms applied to robot learning were presented. Two supervised paradigms were presented: Inductive Concept Learning and Explanation Based Learning, since they require substancial feedback from a teacher to guide the learning process, and two unsupervised learning paradigms: Evolutionary Learning and Reinforcement Learning, that require little to no feedback during the learning process.

REFERENCES

[1] Tom M. Mitchell, *Machine Learning*, WCB/McGraw-Hill, 1997.

[2] Michael D. Schmidt and Hod Lipson, "Learning noise", in *GECCO*, Hod Lipson, Ed. 2007, pp. 1680–1685, ACM.

[3] Sebastian Thrun, "Learning occupancy grid maps with forward sensor models", *Autonomous Robots*, vol. 15, no. 2, pp. 111–127, 2003.

[4] Sebastian Thrun, Wolfram Burgard, and Dieter Fox, *Probabilistic Robotics*, MIT Press, 2005.

[5] Martin Riedmiller, "Machine learning for autonomous robots", in *KI*, Susanne Biundo, Thom W. Frühwirth, and Günther Palm, Eds. 2004, vol. 3238 of *Lectures Notes in Computer Science*, pp. 52–55, Springer.

[6] Sridhar Mahadevan, "Machine learning for robots: A comparison of different paradigms", in *Workshop on Towards Real Autonomy , IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS-96)*, 1996.

[7] "Inductive bias - wikipedia", `http://en.wikipedia.org/wiki/Inductive_bias`, Accessed in December 2010.

[8] Dean Pomerleau, "Alvinn: An autonomous land vehicle in a neural network", in *NIPS*, David S. Touretzky, Ed. 1988, pp. 305–313, Morgan Kaufman.

[9] David Stavens, Gabriel Hoffmann, and Sebastian Thrun, "Online speed adaptation using supervised learning for high-speed, off-road autonomous driving", in *IJCAI*, Manuela M. Veloso, Ed., 2007, pp. 2218–2224.

[10] Pieter Abbeel, Dmitri Dolgov, Andrew Y. Ng, and Sebastian Thrun, "Apprenticeship learning for motion planning with application to parking lot navigation", in *IROS*. 2008, pp. 1083–1090, IEEE.

[11] Douglas Vail and Manuela Veloso, "Learning from accelerometer data on a legged robot", in *Proceedings of the 5th IFAC/EURON Symposium on Intelligent Autonomous Vehicles*, 2004.

[12] Sebastian Thrun, Michael Montemerlo, Hendrik Dahlkamp, David Stavens, Andrei Aron, James Diebel, Philip Fong, John Gale, Morgan Halpenny, Gabriel Hoffmann, Kenny Lau, Celia M. Oakley, Mark Palatucci, Vaughan Pratt, Pascal Stang, Sven Strohband, Cedric Dupont, Lars-Erik Jendrossek, Christian Koelen, Charles Markey, Carlo Rummel, Joe van Niekerk, Eric Jensen, Philippe Alessandrini, Gary R. Bradski, Bob Davies, Scott Ettinger, Adrian Kaehler, Ara V. Nefian, and Pamela Mahoney, "Stanley: The robot that won the darpa grand challenge", *Field Robotics*, vol. 23, no. 9, pp. 661–692, 2006.

[13] Tom M. Mitchell and Sebastian Thrun, "Explanation based learning: A comparison of symbolic and neural network approaches", in *ICML*, 1993, pp. 197–204.

[14] Stefano Nolfi and Dario Floreano, *Evolutionary Robotics*, MIT Press, 2000.

[15] Hugo Picado, Marcos Gestal, Nuno Lau, Luís Paulo Reis, and Maria Tomé, "Automatic generation of biped walk behavior using genetic algorithms", in *IWANN (1)*, Joan Cabestany, Francisco Sandoval Hernández, Alberto Prieto, and Juan M. Corchado, Eds. 2009, vol. 5517 of *Lecture Notes in Computer Science*, pp. 805–812, Springer.

[16] Hod Lipson, "Evolutionary synthesis of kinematic mechanisms", *AI EDAM*, vol. 22, no. 3, pp. 195–205, 2008.

[17] Richard S. Sutton and Andrew G. Barto, *Reinforcement Learning: An Introduction*, MIT Press, 1998.

[18] Martin Riedmiller, "Neural fitted Q iteration first experiences with a data efficient neural reinforcement learning method", in *In 16th European Conference on Machine Learning*. 2005, pp. 317–328, Springer.

[19] Thomas Gabel, Sascha Lange, Martin Lauer, and Martin Riedmiller, "Bridging the gap: Learning in the robocup simulation and midsize league", in *In Proceedings of the 7th Portuguese Conference on Automatic Control (Controlo*, 2006.

[20] Heiko Müller, Martin Lauer, Roland Hafner, Sascha Lange, Artur Merke, and Martin Riedmiller, "Making a robot learn to play soccer using reward and punishment", in *KI*, Joachim Hertzberg, Michael Beetz, and Roman Englert, Eds. 2007, vol. 4667 of *Lecture Notes in Computer Science*, pp. 220–234, Springer.

[21] "Robocup multi-agent learning special interest group", `http://sserver.sourceforge.net/SIG-learn/node1.html`, Accessed in December 2010.

[22] "A robocup four legged league game", `http://www.garcia-camino.es/img/robocup.jpg`, Accessed in December 2010.