Journal of Computing and Information Technology - CIT 16, 2008, 4, 247–254
doi:10.2498/cit.1001392

An Assessment of Machine Learning Methods for Robotic Discovery

Ivan Bratko
Faculty of Computer and Information Science, University of Ljubljana, Slovenia

In this paper we consider autonomous robot discovery through experimentation in the robot's environment. We analyse the applicability of machine learning (ML) methods with respect to various levels of robot discovery tasks, from extracting simple laws among the observed variables, to discovering completely new notions that were never mentioned in the data directly. We first present some illustrative experiments in robot learning in the XPERO European project. Then we formulate a systematic list of types of learning or discovery tasks, and discuss the suitability of chosen ML methods for these tasks.

Keywords: machine learning, robotic discovery, autonomous learning, gaining insights

1. Introduction

In this paper we analyse the applicability of machine learning (ML) methods in various robot discovery tasks, in particular in the light of the experience in the XPERO project. We first introduce the XPERO project. Then we formulate a list of types of XPERO learning or discovery tasks, and of applicable ML methods, and we discuss the suitability of one against the other.

XPERO is a Sixth Framework European project with full title Learning by Experimentation (www.xpero.org). The scientific goal of XPERO is to investigate the mechanisms of autonomous discovery through experiments in an agent's environment. In XPERO, the experimental domain is the robot's physical world, and the subject of discovery are various, quantitative or qualitative laws in this world. Discovery of such laws of (possibly naive) physics would enable the robot to make predictions about the results of its actions, and thus enable the robot to construct plans that would achieve the robot's goals.

In addition to discovering laws that directly enable predictions, XPERO aspires to advance the understanding of the mechanisms that enable *new insights*. In XPERO, an "insight" means something conceptually more general than a law. One possible definition of an insight in the spirit of XPERO is the following: an insight is a new piece of knowledge that makes it possible to simplify the current agent's theory about its environment. Examples of insights are the discoveries of notions like absolute coordinate system, arithmetic operations, notion of gravity, notion of support between objects, etc. An insight would thus ideally be a new concept that makes the current domain theory more flexible and enables more efficient reasoning about the domain. Thus, insights should also make further learning in this domain easier and more effective.

It should be noted that the scientific goals of XPERO are considerably different from the goals of a typical robotics project. Therefore, also the criteria of success in XPERO are different from typical criteria in robotics. In a typical robotics project, the goal may be to improve the robot's performance at carrying out some physical task. To this end, any relevant methods, as powerful as possible, will be applied. In contrast to this, in XPERO we are less interested in improving the robot's performance than in making the robot gaining insights, that is improving the robot's theory and "understanding" of the world. We are interested in finding mechanisms, as generic as possible, that enable the gaining of insights. For such a mechanism to be generic, it has to make only a few rather basic assumptions about the agent's prior knowledge.

We are interested in minimizing such “innate knowledge” because we would like to demonstrate how discovery and gaining insights come about from only a minimal set of “first principles”. In addition, as our aim is gaining insights, not all machine learning methods are appropriate. Our aim requires that the induced insights can be interpreted and understood, and are not only useful for making predictions. Therefore, new insights have to be represented in *explicit symbolic* form. This requirement largely rules out some otherwise effective machine learning methods, such as neural networks and support vector machines.

It is envisaged that gaining insights will require sophisticated mechanisms, such as iterative completion of the “experimental loop”. Experimental loop comprises the following activities: determining next goals, planning experiments that serve attaining these goals, collecting measurement data from these experiments, and analysis of experimental data. Data analysis will typically involve methods of data mining and machine learning, which will also give rise to the formation of new hypotheses.

It is expected that standard techniques of machine learning (ML) and data mining will have to be adapted to the specific needs of discovery in XPERO. However, we investigate in this paper what can be expected to be achieved, in terms of above stated robot discovery, from straightforward application of standard machine learning techniques to experimental data collected by the robot. In the sequel, we first describe some initial experiments and give some illustrative results. Then we specify more systematically a number of generic discovery tasks in XPERO, a repertoire of ML approaches, and analyze these approaches according to their advantages and shortcomings with respect to the tasks.

2. Setup in Initial Experiments

In initial ML experiments in XPERO, we used a simple experimental setup, called “movability”. This setup consisted of a mobile robot moving in the plane with just one additional object, a red block or ball. The robot, equipped with stereo vision capability, could detect the area in the image belonging to the object, robot’s distance from the object and the angle between the current orientation of the robot and the angle at which the object was observed. These data were sampled in time at discrete time steps. Also, the robot was aware of its actions, that is moves, expressed as the step distance in the forward direction (i.e. its current orientation), and the “step angle”, that is the change of the robot’s orientation. Figure 1 illustrates these parameters.

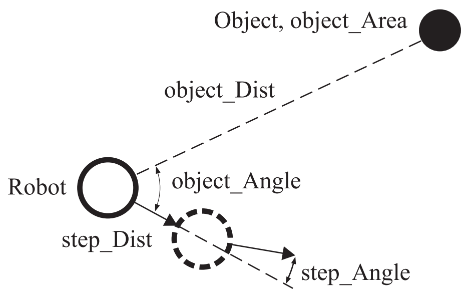

Figure 1. Movability experiments: relations between the robot and the object, and between the current and the next robot’s positions. The arrows show the robot’s orientation, that is the current direction of robot’s movement.

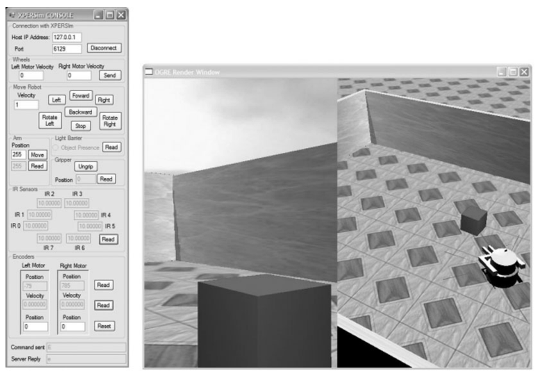

Figure 2. A snapshot of the XPERsim simulation of Khepera II robot.

The experiments according to the setup shown in Figure 1 were carried out by both a robot simulator (XPERsim, Figure 2) and various real robots (e.g. Poineer D2X). XPERSim [1] is a robot simulator designed to provide a common simulation system for the XPERO consortium. It enables the replication of experiments, regardless of the physical presence of the robot and speeding up the pace of research. XPERSim is designed to simulate different mobile platforms. One of these platforms is the Khepera II robot from the K-TEAM Switzerland, used in some experiments in this paper.

Pioneer D2X robot was used to run the movability experiments physically [4]. The robot was equipped with odometry, eight sonar sensors and a stereo vision system. The single object used in the experimental setup was a red ball (radius 50mm). Another camera from the top of the scene was used to provide a separate set of data using "world's coordinate system" which was only meant to be used for verification and evaluation of learning methods and not for machine learning. According to the philosophy of using a minimal set of "first principles", these experiments always assume that the robot is not aware of the absolute coordinate system.

The "observation vector" of the robot at some time point consists of the following variables (see Figure 1):

1. *object_Dist*, the distance to the object
2. *object_Angle*, the angle of the object w.r.t. current robot's orientation
3. *object_Area*, the perceived size of the object as it appears in the image from robot's camera
4. *step_Dist*, the distance traveled in the last time step (zero, if the robot is turning in place)
5. *step_Angle*, the change of angle in the last step (zero, if the robot is moving along straight line)

In the learning experiments we learned both "static" and "dynamic" relations. A static relation defines the relation between the variable values at the same time point. An example of a static relation is the function that maps *object_Dist* and *object_Angle* into *object_Area*, where all these variable values are taken at the same time point t. The corresponding learning problem is to induce, from the given data, a function f for prediction of perceived object area in the camera image from the distance and angle to the object:

$$object\_Area = \mathrm{f}(object\_Dist,\ object\_Angle)$$

In this learning problem, in the usual terminology of ML, *object_Dist* and *object_Angle* are called the attributes, and *object_Area* is called the class. Dynamic relations are useful to make predictions about variable values at the next time point $t + 1$ (assuming that time points are indexed by integers) from the given variable values at time t and robot's actions at time t. An example of learning dynamic relations is to induce a function g that predicts object_Area at time $t + 1$:

$$\begin{aligned} &object\_Area(t+1) = \\ &\quad \mathrm{g}(object\_Dist(t),\ object\_Angle(t), \\ &\qquad step\_Dist(t),\ step\_Angle(t)) \end{aligned}$$

3. Machine Learning Techniques Used

We experimented with a number of different ML techniques chosen so that they constitute a representative sample of ML approaches. For reasons explained in the introduction, our choice of ML techniques was limited to those that induce concept descriptions in explicit symbolic form. The chosen techniques range from the well known ones to the less widely known. The wellknown techniques used were: induction of decision trees, regression trees, and induction of if-then rules. The less known methods include the learning of equations (system GoldHorn) and the learning of qualitative models (systems QUIN and Padé). Most of the implementations of these techniques that we used in the experiments are integrated in the Orange ML environment [6]. In the following paragraphs, we briefly describe these techniques and their implementations.

Rule learning using CN2. CN2 is a rule induction algorithm that can cope with noisy data [5]. We experimented with the implementation of CN2 in Orange. The output of this algorithm is an ordered set of if-then rules, also known as "decision lists".

Induction of classification and regression trees. In a classification tree (also called decision tree), the internal nodes correspond to selected attributes, and the branches that originate at a node correspond to the values of the node's attribute. The leaves of the tree give the classification that applies to all instances that reach the leaf (i.e. the cases whose attribute values are in agreement with the attribute values along the path from the root of the tree to the corresponding leaf). In the case of noisy data, for example, the classification takes the form of a probability distribution over all possible classes. In classification trees, the class is discrete. Regression trees are similar to classification trees, except that the class variable is real-valued. Regression tree learning is more directly applied in our domain where all the variables are continuous. However, classification tree learning is also relevant after discretizing a continuous class variable into intervals. We used the Orange implementation of classification and regression tree induction.

Equation induction with GoldHorn. GoldHorn [8,9] is a machine learning system intended to discover empirical laws, in the form of algebraic equations, that govern the behavior of dynamic systems. Given a behavior of the system, e.g. a sampled execution trace, GoldHorn attempts to find a set of ordinary differential equations that describe the dynamics of the system. GoldHorn does not simply fit the parameters of equations of given forms, but it also constructs new forms of equations. The significance of an equation is judged by the degree of fit with the data (correlation coefficient and normalized squared error). It should be noted that, as a result, GoldHorn not only finds differential equations of some predefined forms, but also does *structural* synthesis of new forms of equations.

Learning qualitative models with QUIN. QUIN (Qualitative Induction [12]) is a learning program that looks for qualitative patterns in numerical data. QUIN expresses such qualitative patterns by *qualitative trees* that are similar to decision trees, but have monotonic qualitative constraints in the leaves. *Monotonic qualitative constraints (MQCs)* are a kind of monotonicity constraints that are widely used in the field of qualitative reasoning and are a generalization of monotonic function constraint used in QSIM [10]. For example, the MQC $Z = \mathrm{M}^{+,-}(X, Y)$ says that Z is monotonically increasing in X and monotonically decreasing in Y. If both X and Y increase, then according to this constraint, Z may increase, decrease or stay unchanged. In such a case, an MQC cannot make an unambiguous prediction of the qualitative change in Z.

Learning qualitative models with Padé. Padé [14] is a set of methods for estimating quantitative or qualitative partial derivatives for points in an attribute space when given a set of learning examples in this space. The name Padé is an acronym for "partial derivative" (and also the name of a famous French mathematician). Once qualitative derivatives are extracted with Padé, one can use arbitrary machine learning algorithms (e.g. C4.5, CN2, Naïve Bayes, SVM etc.) to induce a corresponding qualitative prediction model.

The Orange environment. Orange [6] is a library of C++ core objects and routines that includes a large variety of machine learning and data mining algorithms, plus routines for data input and manipulation. It is a collection of Python-based modules that sit over the core library and offer a versatile environment for developers, researchers and data mining practitioners. Orange also comprises a set of graphical widgets that use methods from the core library and Orange modules and provide a nice user interface. All these make Orange a comprehensive, component-based framework for machine learning and data mining which is constantly growing by adding more and more machine learning methods and algorithms while at the same time improving the existing ones.

4. Some Experimental Results

Among many results in learning various static and dynamic relations in the movability scenario with different ML techniques, we here select some examples to highlight some positive and negative points. In general, more interesting results were obtained with the learning of qualitative models, both with QUIN and Padé, and with HYPER with respect to discovery of new concepts (effectively extending the description language).

Here is a qualitative tree, written as a nested if-then rule, for the class variable's *object_Area* qualitative dependence on *object_Dist* and *object_Angle*, induced by QUIN from the XPER-Sim Khepera traces:

```
if $object\_Angle \leq 2.6769$
    then if $object\_Angle \leq -13.76$
        then $\mathrm{M}^{+,-}(object\_Angle, object\_Dist)$
        else $\mathrm{M}^{-}(object\_Dist)$
else if $object\_Angle \leq 8.4$
        then $\mathrm{M}^{-}(object\_Dist)$
        else $\mathrm{M}^{+,-}(object\_Angle, object\_Dist)$
```

When the robot is facing away from the object, the object is not visible, and the area is always zero. The tree above was induced only from examples where the object is visible. The middle two leaves of the tree that apply when the robot is facing the object (angle between -13.76 and 8.4 degrees), say that the area decreases with increasing distance. This is what one would expect, as the perceived object becomes smaller when we move away. The leftmost leaf and the rightmost leaf describe similar decreasing dependence of area on object distance, but something else is also going on there: the disappearing of the object from the camera view when the robot is turning away from the object.

The induced qualitative tree gives a good qualitative explanation of how the area changes with object distance and object angle. Non-symmetry in the angular thresholds in the left and right sub-trees is a blemish due to statistical asymmetry in the data between positive and negative angles. This qualitative tree could be viewed as a first simple discovery. It however falls short of being an insight in the particular ambitious sense of XPERO discussed in Section 1.

The same data was used for learning with GoldHorn. In this case, when static relations were considered only, GoldHorn generated algebraic equations (not differential). The first problem is that GoldHorn induces hundreds of equations and many of them, even those with the highest rank, are useless or meaningless. Typically, such equations are the result of spurious correlations between the variables and numerical errors. One such spurious and meaningless equation is equation 3 below. The highest ranked equations according to the squared error were:

(1) $\text{object\_Area} * \text{object\_Dist} = 99.19 + 11.48\ \text{object\_Area}$

(2) $\text{object\_Area}^2 * \text{object\_Dist} = -463.49 + 299.99\ \text{object\_Area}$

(3) $\text{object\_Area}^2 = -62.04 + 19.98\ \text{object\_Area}$

(4) $\text{object\_Area} * \text{object\_Dist}^2 = 6232.31 - 6.37\ \text{object\_Angle}^2$

(5) $\text{object\_Area}^3 = -470.17 + 19.37\ \text{object\_Area}^2$

The first three equations do not mention object angle and obviously cannot serve as valid models. The third equation and the fifth equation are meaningless. The only interesting equation is the fourth equation that can be rewritten as:

$$\text{object\_Area} = (6232.31 - 6.37\ \text{object\_Angle}^2)\ /\ \text{object\_Dist}^2$$

Similar to the above qualitative tree, it says that the area decreases with object distance. Actually, this equation says more – it correctly states that the area decreases with the square of distance. The equation also indicates the decreasing of the area when a part of the object is disappearing from the camera. However, this is not correct for small object angles, i.e. when the whole object is visible by the camera.

The learning with Padé also produced sensible results, of similar form as those by QUIN.

The experiments with the learning of dynamic relations showed several ways of the learning systems going astray, and were overall quite disappointing. This is surprising and no obvious explanation for this was found. Therefore, I suspect that different reformulations of the dynamics learning prolem could be more successful.

The induced models presented so far (described in more detail in [4]) are concerned with prediction where no new concept was introduced. These may be useful for making predictions, but do not count as “insights” in the more ambitious sense of XPERO described in the Introduction. Therefore, in the strict, abstract sense of “gaining insights” in XPERO, more interesting results were obtained with another learning paradigm, Inductive Logic Programming (ILP) in a somewhat different experimental scenario. Here the robot was pushing blocks in a two-dimensional space. The robot was able to actually move some of the blocks, but some of the blocks could not be moved. After some time, the robot collected a number of experimental data recorded as ground facts about predicates: at(Obj,T,P), meaning object Obj was observed at position P at time T; move(Obj,P1,D,P2), meaning command “move Obj from P1 by distance D” resulted in Obj at P2. Here all positions are two-dimensional vectors. The robot’s prior knowledge consisted of predicates: different(X,Y), meaning X and Y are not approximately equal; add(X, Y, Z), meaning $Z \approx X + Y$. These relations are defined as approximations because of noise in numerical data. It should be noted that neither the observations nor the prior knowledge contain the concept of mobility, or any mention of it. There are no examples given of movable and not movable objects. The ILP program HYPER [2] was used on this learning problem to induce a theory of moving in this world (that is learn predicate move/4). The induced theory by HYPER was stated in logic by the following Prolog clauses:

```
move(Obj,Pos,Dist,Pos):-
  not p(Obj).

move(Ob,Pos1,Dist,Pos2):-
  add(Pos1,Dist,Pos2),
  p(Obj).

p(Obj):-
  at(Obj,T1,Pos2),
  at(Obj,T2,Pos2),
  different(Pos1,Pos2).
```

In the clauses above, the variables were renamed for easier understanding. The first clause deals with unmovable objects (after the move command, the position of the object remains unchanged). The second clause handles movable objects. The point of interest is that HYPER invented a new predicate, p(Object) which is true for objects that can be moved. The definition above says that an object is movable if it has been observed at two different positions in time. Thus the robot has come up with a new concept never mentioned in the data or problem definition. The new concept p(Object) enabled the learning system to divide the problem in the two cases.

5. Robot Discovery Tasks and Learning Paradigms

Now we formulate a more comprehensive set of XPERO types of learning and discovery tasks that go beyond the illustrations in the previous section:

(1) *Learning static functional relations*. Given a set of observations, all taken at the same time t, find relations between them. This usually consists of finding mappings from a subset of observations into other observations. Examples in the movability scenario were discussed in the previous section, such as object area being a function of distance and angle. It should be noted, however, that in this example setting, this relation is only functional for the cases when the angle is determined for all possible orientations of the robot. This may not be the case for example when the distance is determined through stereo vision (and not by proximity sensors).

(2) *Learning dynamic relations* (learning the dynamics of the robot's world). These relations between observations, states and actions may be expressed in many ways, such as:

- Obs(t + 1) =f(Obs(t),Act(t)) (this assumes that current observations are sufficient to determine complete consequences of any action and next observations
- State(t + 1) = f(State(t),Act(t)), Obs(t) = g(State(t)) This assumes that the state of the system is given
- Same as above, but the state of the system is not known, or given (observable), but has to be discovered e.g. as State(t) = f(History(t)) where History(t) = (Obs(t−1), Act(t−1), Obs(t−2), Act(t−2), . . .). The learning also needs to discover a suitable length of History. This may give rise to some insights, such as: (a) discovery of the concept of state, and (b) given that the robot assumes the concept of state, it then has to discover its definition in terms of its relation (f above) with observations and actions.

(3) *Discovering useful notions in the physical world* of the robot, such as: movable(Object), obstacle(Object, Robot, Intention), containment, force, gravity, friction, support, touch, push, . . .

(4) *Discovering aggregates*: for example stack of blocks, "train" of blocks

(5) *Discovering notions of cooperation* of several agents, e.g. two robots jointly pushing a long block

(6) *Discovering functional concepts*: e.g. a long block used as a tool to push a number of smaller blocks in parallel

Machine learning paradigms considered in this analysis include the following:

(1) Attribute-value learning, such as learning if-then rules, decision trees, regression trees and regression rules.

(2) Induction of equations from numerical data (programs like Goldhorn and Lagramge [7]).

(3) Qualitative learning, like programs QUIN and Padé. Hypothesis languages in this learning paradigm express qualitative relations in terms of qualitative proportionalities or monotonicity constraints among variables in the system.

(4) Q^2 learning (also called qualitatively faithful quantitative learning [13]) possibly realised as a combination of qualitative learning with QUIN or Padé, and qualitatively constrained regression e.g. with Qfilter or QCgrid. This is a non-standard approach to attribute-value learning with numerical class variable whereby the induced regression model respects the qualitative properties detected in the learning data.

(5) Inductive logic programming (ILP) with programs like HYPER, Progol and Aleph. In addition to the learning of logical theories with ILP, we here consider also the learning of dynamic qualitative models [3] of QSIM-type since such models can be naturally represented by logic clauses, as well as simulated by logic programs.

6. Applicability of Learning Methods to Discovery Tasks

The learning paradigms above are roughly ordered according to the power of representation languages, and, correspondingly, according to increasing computational complexity. So the range of applicability of these paradigms is on the one hand limited by expressiveness of hypothesis languages, and on the other hand by computational complexity. Another limiting factor is the interpretability (understandability) of induced descriptions. This ability we consider as necessary in XPERO with respect to the gaining of insights. For this reason, we do not seriously consider the use of some other ML approaches, such as neural networks, support vector machines, and reinforcement learning in its usual form.

Table 1 summarises the suitability of the considered ML approaches for various types of learning/discovery tasks in XPERO. The applicability is assessed in terms of four criteria:

1. Expressiveness (E) of the ML method's hypothesis language (representation)
2. Predictive accuracy (A) of induced theories; this is to be understood as rough estimate that also takes into account the ability to cope with noise, precision in numerical prediction etc.
3. Computational complexity (C)

	Learning static relations				Learning dynamic relations				Discovering physical notions				Discovering aggregate notions				Discovering agent cooperation				Discovering functional notions			
	E	A	C	I	E	A	C	I	E	A	C	I	E	A	C	I	E	A	C	I	E	A	C	I
Att.value learning		+	+			+	+		−				−					+	+		−			
Equation learning	+				+	+			−				−				−				−			
Qualitative learning	+	−	+	+	+	−		+									+			+				
Q^2 learning	+	+	+	+	+	+		+									+	+		+				
ILP	+	−	−	+	+	−	−	+	+		−	+	+		−	+	+		−	+	+		−	+

Table 1. Tentative assessment of applicability of ML aproaches to types of task. Criteria: E = expressiveness, A = accuracy, C = complexity, I = interpretability. "+" means above average, "−" means below average, blank means neutral or not applicable.

4. Interpretability (I) of induced hypotheses; how easy or difficult it is for the user to figure out the meaning of a hypothesis, and possibly use it for reasoning about the domain.

It should be emphasised that Table 1 only gives some general ideas of the comparative advantages of the potentials of the methods considered, and should not be understood as a definite ranking of the methods that applies to all cases and all situations. Several entries in the table are only tentative, and a more firm assessment would require very substantial further work. Hopefully, further work in XPERO will help to clarify some of these judgements.

Acknowledgments

The research described in this paper was supported by the European Commission, 6th Framework project XPERO, and the Slovenian research agency ARRS, Research program Artificial Intelligence and Intelligent Systems. The work described in the section “Initial experiments in XPERO” was carried out by a number of people, including the co-authors of paper [4].

References

[1] I. AWAAD AND B. LEON (2006). XPERSim: Simulation of the robotic experimenter. *XPERO Report on R&D 1.* September 2006.

[2] I. BRATKO (2001) Prolog Programming for Artificial Intelligence, 3rd edition, Addison-Wesley / Pearson.

[3] I. BRATKO AND D. ŠUC (2003 Learning qualitative models. *AI Magazine*, Vol. 24 (2003), no. 4, pp. 107–119.

[4] I. BRATKO, D. ŠUC, I. AWAAD, J. DEMŠAR, P. GEMEINER, M. GUID, B. LEON, M. MESTNIK, J. PRANKL, E. PRASSLER, M. VINCZE, J. ŽABKAR (2007) Initial experiments in robot discovery in XPERO, ICRA’07 Workshop Concept Learning for Embodied Agents, Rome, april 2007.

[5] P. CLARK AND T. NIBLETT (1989) The CN2 induction algorithm. *Machine Learning*, Vol. 3, pp. 262–284.

[6] J. DEMŠAR AND B. ZUPAN (2006). Orange: Data Mining Fruitful & Fun – From Experimental Machine Learning to Interactive Data Mining. `http://www.ailab.si/orange`.

[7] S. DŽEROSKI AND L. TODOROVSKI (1993). Discovering dynamics. In *Proceedings of the 10th International Conference on Machine Learning*, 97–103. Morgan Kaufmann.

[8] V. KRIŽMAN, S. DŽEROSKI AND B. KOMPARE (1995). Discovering dynamics from measured data. *Electrotechnical Review*, 62:191–198.

[9] V. KRIŽMAN (1998). *Automatic Discovery of the Structure of Dynamic System Models.* PhD Thesis, Faculty of Computer and Information Sciences, University of Ljubljana.

[10] B. KUIPERS (1994). *Qualitative Reasoning: Modeling and Simulation with Incomplete Knowledge*, MIT Press.

[11] D. ŠUC (2001). Machine reconstruction of human control strategies, Ph.D. Thesis, Faculty of Computer and Information Science, University of Ljubljana, Slovenia.

[12] D. ŠUC (2003). Machine Reconstruction of Human Control Strategies, in: Frontiers Artificial Intelligence Appl., vol. 99, IOS Press, Amsterdam.

[13] D. ŠUC, D. VLADUŠIČ AND I. BRATKO (2004). Qualitatively Faithful Quantitative Prediction, *Artificial Intelligence, 158:2*, pp. 190–219.

[14] J. ŽABKAR, I. BRATKO AND J. DEMŠAR, Learning qualitative models through partial derivatives by Padé. QR07, 21st International Workshop on Qualitative Reasoning, 2007, Aberystwyth, Wales, U.K., pp. 193–202.

Received: June, 2008
Accepted: September, 2008

Contact address:
Ivan Bratko
University of Ljubljana
Faculty of Computer and Info. Sc.
Tržaška 25, 1000 Ljubljana, Slovenia
e-mail: `bratko@fri.uni-lj.si`

IVAN BRATKO is professor of computer science at Faculty of Computer and Information Science, Ljubljana University, Slovenia. He heads the AI laboratory at the University. He has conducted research in machine learning, knowledge-based systems, qualitative modelling, intelligent robotics, heuristic programming and computer chess. His main interests in machine learning have been in learning from noisy data, combining learning and qualitative reasoning, robotic discovery, and various applications of machine learning including medicine, ecological modelling and control of dynamic systems. Ivan Bratko is the author of widely adopted text PROLOG Programming for Artificial Intelligence (third edition: Pearson Education 2001).