Auto-supervised learning in the Bayesian Programming Framework

Pierre Dangauthier, Pierre Bessiere, Anne Spalanzani

Auto-supervised learning in the Bayesian Programming Framework*

Pierre Dangauthier, Pierre Bessière and Anne Spalanzani
E-Motion
INRIA / GRAVIR - CNRS
655 Avenue de l'Europe, Montbonnot
38334 Saint Ismier cedex - France
pierre.dangauthier@imag.fr

***Abstract*— Domestic and real world robotics requires continuous learning of new skills and behaviors to interact with humans. Auto-supervised learning, a compromise between supervised and completely unsupervised learning, consist in relying on previous knowledge to acquire new skills. We propose here to realize auto-supervised learning by exploiting statistical regularities in the sensorimotor space of a robot. In our context, it corresponds to achieve feature selection in a Bayesian programming framework. We compare several feature selection algorithms and validate them on a real robotic experiment.**

***Index Terms*— Auto-supervised learning, Uncertain Environment, Feature Selection, Bayesian Programming, Genetic Algorithms.**

I. Introduction

In a real environment, a robot needs to continuously improve its knowledge to interact with humans. It needs to better the performance of its previously known skills and to learn new ones. In a complex environment, learning totally new behaviors probably require human feedback. But there are simple situations where unsupervised, or more precisely auto-supervised learning, is possible. We study in this article the case of a robot which improves the use of its body without any human supervision. In a simple tracking task, the robot learns by itself to use its laser sensor (SICK) instead of its camera. This is done by searching for correlations in the space of its sensor and motor variables during the tracking behavior.

A. Auto-supervised learning

Auto-supervised learning is biologically inspired and is strongly related to mental development [1]. This way of learning takes place during baby's development, but also during adulthood. For instance, a beginner strongly relies on his sight to use a computer keyboard. Then, he learns to use his finger sensibility and spatial position to perform the same task. One can say that he learned a new sensorimotor behavior thanks to the supervision of his sight.

*This work is partially supported by the European Bayesian Inspired Brain and Artefacts (BIBA) project, by the French National Institute for Research in Computer Science and Control (INRIA), and by the French Research Ministry.

This learning is possible by searching, and exploiting statistical regularities in the sensorimotor space. The exploitation of statistical regularities can be seen as the foundation of learning and is a promising model of cognition [2]. Finding regularities means finding compact data representations, with which a system becomes able to generalize and makes sense of its perceptions.

B. Bayesian Robot Programming

Apart from the notion of auto-supervised learning, a domestic robot perceives abundant, uncertain and often contradictory information. Sensors and actuators are not perfectly reliable. Therefore, classical determinist programming has been shown to be unable to address real world problems [3]. We prefer to use a probabilistic approach method called **Bayesian Programming**. Bayesian Robot Programming is a promising candidate in this context [4] and has given several interesting results [5] [6]. Moreover, Bayesian Inference is a promising model for understanding animal perception and cognition [7].

Bayesian Robot Programming is based on the subjective interpretation of probabilities deeply described by E.T. Jaynes [8]. A Bayesian program is a probabilistic representation of the relations between sensors and actuators in order to perform a specified task.

In this framework, a Bayesian programmer starts to describe a task by specifying **preliminary knowledge** to the robot. Then the robot processes Bayesian inference in order to take a decision regarding its inputs.

Establishing preliminary knowledge can be divided into three parts:

- Define relevant variables for the problem;
- Possibly assume conditional independencies between them;
- Choose prior distributions on those variables.

The reader can find detailed examples of non-trivial Bayesian programs in [5].

C. Goal

In this work, we present an attempt to automate the creation of a Bayesian program. The "relevant variables"

part of a new program will be autonomously discovered under the supervision of another Bayesian program.

The initial behavior of the robot is to track a red ball with a digital camera. This behavior is controlled by an initial Bayesian program. During a tracking experiment, all sensor inputs are recorded, including the direction of the ball. Then, the robot looks for sensors highly correlated with the position of the ball. After that, the robot builds a new tracking program with these sensors, and is then able to stare at the target without its camera.

We present here several algorithms for discovering the relevant variables related to that simple tracking task.

D. Approach

Once the robot has recorded, for each time step, the position of the ball given by its camera and all other sensor values, the problem is reduced to classical supervised learning. Thus, the goal is to find the minimal set of variables allowing a good prediction of the position.

Finding relevant variables for good predictions is a well known problem in the AI field. In our case, the selected subset of variables will be the input of a naive Bayesian classifier [9]. This search is therefore called **feature selection for machine learning**.

This article presents different feature selection methods for finding correlations between a variable and the sensor data. We show that our methods enhance the computing time and the recognition rate of the classification. A study of the selected sensors allows us to validate the relevance of our approach regarding the investigation for new sensorimotor modalities.

II. Experiment

A. Presentation

The problem is as follows: we have a robot with a lot of different sensors (laser range SICK, proximeters, thermometer, odometers, battery level...) and a ball is moving in the horizontal plane. The ball position is defined by the angle θ between the ball and the robot's main axe. A set of learning examples is recorded. For each time step, we record θ and the related values of all the sensors. After a learning stage, our robot should be able to guess the position of the ball knowing its sensor values.

The new constructed Bayesian program will be:

- *Relevant variables* are sensors carrying information about θ. We denote sensors variables by X_i, $\forall i \in [0 \dots N-1]$.
- *Decomposition*: The decomposition of the joint probability distribution we have chosen is called **naive Bayes model**. In our framework, it is reasonable to assume that sensor values are conditionally independent given the value of θ. Thus the decomposition is:

$$P(X_0 \dots X_{N-1},\ \theta) = P(\theta) \prod_{i=0}^{N-1} P(X_i|\theta).$$

- *Priors*: We choose Laplace's histograms [8]. Laplace's histograms, which are simple histograms with non-zero values, are frequently used to model discrete probability distributions.
- *Question*: Once this program is defined, the goal for the robot is to infer the position of the ball θ :

$$\begin{aligned} \theta &= \text{Argmax } P(\theta|X_0 \dots X_N) \\ &= \text{Argmax } \frac{P(X_0 \dots X_N \theta)}{P(X_0 \dots X_N)} \\ &= \text{Argmax } \prod_{i=0}^{N} P(X_i|\theta). \end{aligned}$$

B. Robotics

We carried out our experiments on the **BibaBot** (see Fig.1), the robot of the **BIBA European project** [7]. It is a middle sized (1 - 0.5 - 0.5 m) wheeled robot equipped with a lot of different sensors. In this work, the robot stay motionless.

Fig. 1. *The BibaBot Robot.*

It is equipped with:

- A Pan-Tilt camera,
- A laser scanner (SICK LMS 200 Fig.2) that scans its surroundings in 2D. It gives 360 measures of distance in the horizontal plane. We will consider each laser beam as an individual sensor.
- 3 odometry sensors,
- 4 bumpers,
- 8 ultrasonic sensors,
- 15 infrared proximity sensors.
- a clock and the battery voltage.

The experiment is then done in three steps:

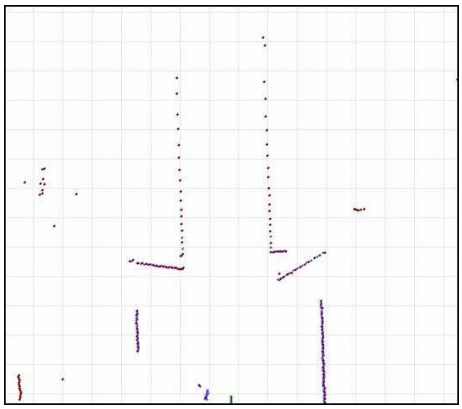

Fig. 2. *A SICK output, when the robot is placed in a corridor*

- Firstly the visual tracking program is launched and a red ball is moving in front of the robot. The program makes the Pan-Tilt camera follow the ball. Thus, we can record θ as the angle of the Pan axis. In the same time we record all the other sensor values.
- Secondly, our feature selection algorithms are launched off line on the collected data. For a given algorithm, a subset of relevant sensors is found.
- Finally we launch the new Bayesian program and see if the robot can guess the position of the ball, or of another object, without any visual information from its camera.

The recognition rate of the position determines the quality of the sensor selection algorithm.

C. Validation criteria

We have different criteria to judge the pertinence of our sensor selection algorithms. The first one is obviously the recognition rate of the classification. Is the robot able to locate the ball with the generated subset of variables? We have also to take into account the computing cost of feature selection, which is crucial for embedded mobile robotics. Another important criterion is the size of the final subset. We aim at minimizing it, while keeping a good recognition rate. The recognition rate is not a monotonic function of the subset size.

Beyond those criteria, we have also to consider the number and the nature of free parameters in our algorithms. In a context of autonomy, the part of the programmer should remain minimal.

III. State of the art

Feature selection is an active field in computer science, especially for *data mining*. Feature selection for machine learning consists in finding the "best" subset of features (sensors for us) for a classification algorithm. Selecting a subset of features before classifying has several advantages:

- It reduces the dimensionality of the problem allowing the application of more complex algorithms,
- It leads to a simpler model of data (Occam Razor argument [10]),
- It enhances the classifier performance, speed and generality,
- It leads to a more comprehensible model and shows conditional dependencies between variables.

Usually feature selection algorithms remove independent or redundant variables.

A. General feature selection structure

It is possible to derive a common architecture from most of the feature selection algorithms (see Fig. 3). These algorithms create a subset, evaluate it, and loop until an ending criterion is satisfied [11]. Finally the subset found is validated with the classifier algorithm on real data.

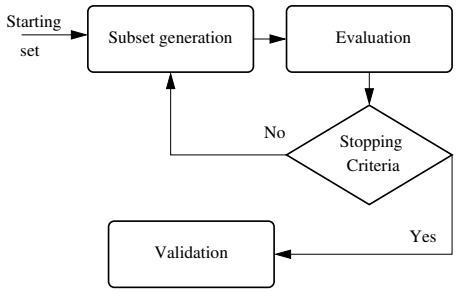

Fig. 3. *General feature selection structure*

1) Subset Generation: The subset generation stage is a search procedure in the space of all subsets. As the size of this space is 2^N, exhaustive search methods are often helpless. Non deterministic search like evolutionary search is often used [12] [13]. It is also possible to employ a heuristic function to build the subsets. There are two main families of heuristic search methods: *forward addition* [14] (starting with an empty subset, we add features after features by local search) or *backward elimination* (the opposite). Ref. [14] presents theoretical arguments in favor of backward elimination, but experimental results [15] shows that forward addition and backward elimination are equivalent. The reader can found in [11] a detailed nomenclature of feature selection algorithms.

B. Subset Evaluation

A simple method for evaluating a subset is to consider the performance of the classifier algorithm when it runs with that subset. This way, the classifier algorithm is wrapped in the loop, and the method is classified as a *wrapper*. On the contrary, *filter* methods do not rely on the classifier algorithm, but use other criteria based on correlation notions.

1) Wrappers: Wrappers have been introduced by John, Kohavi and Pfleger in 1994 [15]. Usually, subset evaluations are a compromise between the learning performance and the number of kept features. Thus wrappers generate

well suited subsets for recognition tasks because they take into account intrinsic bias of the learning algorithm. Another advantage is their conceptual simplicity. There is no need to really understand causalities in data, they only require generating and testing subsets.

But wrappers have several drawbacks. Firstly they do not clarify conditional dependencies between variables, providing theoretical justification for keeping this or this variable. Since selected subsets are specific to a given classifier algorithm, if it is changed, the selection is not valid anymore. More important, wrappers are computationally costly, so this approach may become intractable.

2) Filters: Filters are quicker and based on theoretical notions. But they often give slightly worse recognition rates. In order to rank a subset, a naive solution consists in giving a mark to each variable independently of others, and to sum those scores. Such a feature ranking method can be done by evaluating correlation between a variable and θ. But [16] exposes simple examples showing that this is clearly insufficient. For instance, this can not eliminate redundant variables if they are highly correlated with the target.

In contrast an elegant solution is to consider a subset as a whole, as done in the structural learning method for Bayesian networks [17]. In our case, this can be reduced to the search of Markov blankets [14].

But those theoretical methods are NP-complete and only approximations are implemented.

For those reasons, intermediate methods based on statistical analysis [18] have proven their efficiency [19]. The idea of Ghiselli is to give a good mark to a subset if its variables are highly correlated with the target, but slightly correlated between each other. This is summarized in the following formula:

$$r_{\theta S} = \frac{k\overline{r_{\theta i}}}{\sqrt{k + k(k-1)\overline{r_{ij}}}},$$

where $r_{\theta S}$ is the score of the subset, $\overline{r_{\theta i}}$ is the average correlation between the k variable and θ, and $\overline{r_{ij}}$ is the average of the k^2 intra-correlations. This formula is an approximation in the way that we only consider first order dependencies.

Referring to that method, we need a way to evaluate correlation between two variables. While linear correlation used in [20] is clearly inadequate, it is possible to use classical statistics (like χ^2) estimator [21] [22], or an information based measure. Some authors have explored pure Bayesian independence tests [23] [24], while others rely on other notions as consistency [25]. Recent works try to combine filters and wrappers [16].

C. Stopping criteria and validation

Different kinds of stopping criteria can be used: a computing time, a number of kept variables, a heuristic evaluation of the last subset or a recognition rate. In robotics, this criterion should not be a free parameter, for ensuring autonomy and plasticity. The validation of the final subset will be done by calling the learning algorithm.

IV. Algorithms

We have implemented and compared nine different algorithms. Eight of them were recombination of state of the art methods, and one is original. We quickly present here the best four algorithms: a filter and a wrapper based on genetic algorithms (*WrappGA* and *FiltGA*), a filter based on Ghiselli formula (*GhiselliFilt*) and another filter based on the conditional independency hypothesis (*CondIndFilt*).

A. WrappGA

A genetic algorithm generates subsets which are ranked according to the performance of the classifier algorithms. It ends as soon as a predefined number of generations has been computed.

B. FiltGA

In this case the subset fitness is computed thanks to Ghiselli formula. Correlation between two variables is estimated in the information theory framework by a cross-entropy estimator. The Kullback-Leibler (KL) [26] distance between two probability distributions defines the mutual information between two variables :

$$I(X_i, X_j) = \sum_{X_i}\sum_{X_j} P(X_i, X_j) log(\frac{P(X_i, X_j)}{P(X_i)P(X_j)}).$$

This is in fact the KL distance between $P(X_i, X_j)$ and $P(X_i)P(X_j)$. Therefore the more independent X_i and X_j, the smaller $I(X_i, X_j)$. Taking into consideration that $I(X_i, X_j)$ is biased in favor of variables which have lots of possible values, we can define the uncertain symmetrical coefficient by:

$$USC = 2[\frac{I(X_i, X_j)}{H(X_i) + H(X_j)}].$$

Where $H(X)$ is the entropy of X's distribution. USC is null for independent variables and equals to 1 when X_i and X_j are deterministically linked.

This algorithm ends when a predefined number of generations is computed.

C. GhiselliFilt

GhiselliFilt is basically the same as *FiltGA*, except that the search method is no more evolutionary, but it is a *forward addition* procedure. We start with an empty set and we add variables one by one. At each step we add the variable which maximizes the score of the candidate subset. The score of a subset is computed with the Ghiselli heuristic, which takes into account correlations with θ and inter correlations.

The algorithm ends when it becomes impossible to add a feature without decreasing the score of the subset.

D. CondIndFilt

In this method, we explicitly consider that sensor variables are independent knowing the position of the ball. Several robotic experiments have proven the validity of this hypothesis. The search procedure is *backward elimination*. We start with the complete set of variables, and we try to remove the less informative features. In order to choose a variable, we compute the symmetrized KL distance Δ between two probability distributions.

$$\Delta(\mu, \sigma) = D_{KL}(\mu, \sigma) + D_{KL}(\sigma, \mu).$$

If P is the original distribution, and P_i is the distribution considering X_i independent of θ, we have:

$$\begin{aligned} P(X_0 \dots X_{N-1}\theta) &= P(\theta) \prod_{k=0}^{N-1} P(X_k|\theta) \\ P_i(X_0 \dots X_{N-1}\theta) &= P(\theta)P(X_i) \\ &\quad \prod_{k=0\ k\neq i}^{N-1} P(X_k|\theta). \end{aligned}$$

Then, we just have to compute, for each candidate variable

$$\begin{aligned} \Delta(P, P_i) = \quad & \sum_{\theta} P(\theta) \sum_{X_i} log(\frac{P(X_i)}{P(X_i|\theta)}) \\ & -(P(X_i) - P(X_i|\theta)). \end{aligned}$$

The main drawback is that the algorithm stops when a predefined number of variables are eliminated.

Remark: we have shown that, under conditional independence, the distance relies only on the "difference" between $P(X_i)$ and $P(X_i|\theta)$, e.g. it is not necessary to look at other variables to decide if a variable is useless.

V. Results

	WrappGA	*FiltGA*	*GiselliFilt*	*CondIndFilt*
Comp. Time	- - -	-	+ +	+ + +
Reco. Rate	- -	- -	+ +	+
#Sensors	- -	+	+ +	parameter
Free Parameters	-	-	+ +	- - -

TABLE I

Strengths and weaknesses of different algorithms in the general case. "Free Parameters" line represents the number of free parameters, and of the difficulty to choose them.

We have tested our algorithms both on simulated and robotic data. Here we present the results of the last four experiments:

- **Exp 1.** The robot learns when a red ball is moved, and the new Bayesian program is tested with the same red ball moved differently.
- **Exp 2.** The robot learns with a red ball, but the new program is tested with a wood cube instead of the ball.
- **Exp 3.** Same as **Exp 1.**, without the SICK sensor.
- **Exp 4.** Same as **Exp 2.**, without the SICK sensor.

Numerical results for **Exp 1.** and **Exp 2.** are presented in Table II. We can notice that:

- *GiselliFilt* shows that just 8 laser beams of the SICK are enough to guess the position of the ball. There are two reasons for that:
  - The SICK is a highly reliable sensor.
  - The number of different values chosen during the discretization of θ was quite low (6 different classes for 180 degrees);
- Surprisingly recognition rates are not smaller in Exp.2 than in Exp.1. This shows that the ball and the cube have a similar signature through the kept sensors;
- Algorithms based on genetic algorithms are too slow to be incorporated in a real time processing;
- *WrappGA* and *FiltGA* do not succeed in finding a good subset. The search space is too huge for them; they do not find that only a few sensors are required, even with a lot of generation and a big population. Although in simulated tests, with a dozen of sensors, they find the global optimum better than other methods. Thus we can not use them when too many sensors are used;
- The recognition rate is improved by selecting features. Indeed making a fusion with non relevant sensors decreases the performances.

Numerical results without SICK are presented in Table III. Removing this sensor involves several consequences:

- As the search space drastically shrinks, genetic search performs better;
- Although the recognition rate among algorithms decreases, a good sensor fusion can find the target quite often;
- Sensors kept in the best subset were mainly I.R. relevant sensors with a few U.S. telemeter.

The relative values of algorithms are presented in table I. This study helped us to choose *GhiselliFilt* as a feature selection algorithm. The *forward addition* search procedure combined with Ghiselli heuristic and an entropic measure of mutual information is a good candidate for real-time feature selection. Indeed, Ghiselli formula is an efficient approximation which highlights sensors highly correlated with the target but slightly correlated with other sensors. Moreover mutual information detects more than only traditional linear correlation.

VI. Conclusion and further work

In this work, we have compared different feature selection algorithms. We have tested some of them to enable a robot to discover autonomously correlations in its sensorimotor domain. In our experiments, the robot found a new way to track a ball, passing from visual to proximity

Experiment		*WrappGA*	*FiltGA*	*GiselliFilt*	*CondIndFilt*	*ALL*
1	#Sensors	196	118	8	8	392
	Rec.rate	0.143	0.143	0.8	0.42	0,143
	Time	110.5	8.61	3.98	0.72	0,07
2	#Sensors	183	112	8	8	392
	Rec.rate	0.14	0.143	1	0.43	0,1432
	Time	100.4	8.94	3,92	0,726	0,07

TABLE II

Results of a simple robotic experiment: This table shows that feature selection hugely improves our Bayesian classifier recognition rate.

Experiment		*WrappGA*	*FiltGA*	*GiselliFilt*	*CondIndFilt*	*ALL*
3	#Sensors	14	4	7	8	32
	Rec.rate	0.66	0.2	0.55	0.51	0,55
	Time	2.73	0.15	0.01	0.006	0,002
4	#Sensors	20	5	7	8	32
	Rec.rate	0.40	0.23	0.36	0.38	0,30
	Time	2.97	0.16	0.01	0,005	0,002

TABLE III

Same experiments without SICK sensor. One can see that genetic algorithms perform better in this case with a smaller search space.

tracking. In this framework, feature selection drastically increases performance and speed of the recognition algorithm. This work is a step toward an automatic search of prior knowledge applied to Bayesian robot programming.

Further work will lead to the integration of the selected algorithm in a permanent auto-supervised learning framework. This learning process should be real-time and parallelized.

REFERENCES

[1] J. Weng, J. McClelland, A. Pentland, O. Sporns, I. Stockman, M. Sur, and E. Thelen. Autonomous mental development by robots and animals. *Science*, (291):599–600, Jan 2001.

[2] H Barlow. The exploitation of regularities in the environment by the brain. *Behavioral and Brain Sciences*, 24(3):602–607, 2001.

[3] Pierre Bessière, Eric Dedieu, Olivier Lebeltel, Emmanuel Mazer, and Kamel Mekhnacha. Interprétation ou description (i) : Proposition pour une théorie probabiliste des systèmes cognitifs sensori-moteurs. *Intellectica*, 26-27:257–311, 1998.

[4] Pierre Bessière and the LAPLACE research Group. Survey: Probabilistic methodology and techniques for artefact conception and development. Technical report, Rapport de recherche INRIA, Janvier 2003.

[5] Olivier Lebeltel, Pierre Bessière, Julien Diard, and Emmanuel Mazer. Bayesian robots programming. *Autonomous Robot*, 16(1):49–79, 2003.

[6] C. Pradalier and P. Bessière. Perceptual navigation around a sensorimotor trajectory. In *Proc. of the IEEE Int. Conf. on Robotics and Automation*, New Orleans, LA (US), April 2004.

[7] BIBA. Bayesian Inspired Brain and Artefacts European Project, 2001-2005. http://www-biba.inrialpes.fr/.

[8] E. T. Jaynes. *Probability Theory : The Logic of Science*. G. Larry Bretthorst, 2003.

[9] Pedro Domingos and Michael J. Pazzani. On the optimality of the simple bayesian classifier under zero-one loss. *Machine Learning*, 29(2-3):103–130, 1997.

[10] Anselm Blumer, Andrzej Ehrenfeucht, David Haussler, and Manfred K. Warmuth. Occam's razor. *Information Processing Letters*, 24(6):377–380, 1987.

[11] H. Liu and H. Motoda. *Feature Selection for Knowledge Discovery and Data Mining*. Kluwer Academic Publishers, 1998.

[12] Jihoon Yang and Vasant Honavar. Feature subset selection using a genetic algorithm. *IEEE Intelligent Systems*, 13:44–49, 1998.

[13] Oliver Ritthoff, Ralf Klinkenberg, Simon Fischer, and Ingo Mierswa. A hybrid approach to feature selection and generation using an evolutionary algorithm. Technical Report CI-127/02, Collaborative Research Center 531, University of Dortmund, Germany, 2002.

[14] Daphne Koller and Mehran Sahami. Toward optimal feature selection. In *International Conference on Machine Learning*, pages 284–292, 1996.

[15] George H. John, Ron Kohavi, and Karl Pfleger. Irrelevant features and the subset selection problem. In *International Conference on Machine Learning*, pages 121–129, 1994. Journal version in AIJ, available at http://citeseer.nj.nec.com/13663.html.

[16] Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 2003.

[17] David Heckerman, Dan Geiger, and David Maxwell Chickering. Learning bayesian networks: The combination of knowledge and statistical data. In *KDD Workshop*, pages 85–96, 1994.

[18] Edwin E. Ghiselli. *Theory of Psychological Measurement*. McGraw-Hill Book Company, 1964.

[19] Mark A. Hall. *Correlation-based Feature Selection for Machine Learning*. PhD thesis, Waikato University, Hamilton, NZ, 1998.

[20] Mark A. Hall. Correlation-based feature selection for discrete and numeric class machine learning. In *Proc. 17th International Conf. on Machine Learning*, pages 359–366. Morgan Kaufmann, San Francisco, CA, 2000.

[21] Sir Maurice Kendall and Alan Stewart. *The Advanced Theory of Statistics, Volume 1, 4th Edition*. Mcmillan Publishing, New York, 1977.

[22] H. Liu and R. Setiono. Chi2: Feature selection and discretization of numeric attributes. In *Proceedings of 7th IEEE Int'l Conference on Tools with Artificial Intelligence*, 1995.

[23] Dimitris Margartitis and Sebastian Thrun. A bayesian multiresolution independence test for continuous variables. In *Uncertainty in Artificial Intelligence: Proceedings of the Seventeenth Conference (UAI-2001)*, pages 346–353, San Francisco, CA, 2001. Morgan Kaufmann Publishers.

[24] Marco Zaffalon and Marcus Hutter. Robust feature selection by mutual information distributions. In *Proceedings of the 14th International Conference on Uncertainty in Artificial Intelligence*, 2002.

[25] Hussein Almuallim and Thomas G. Dietterich. Learning boolean concepts in the presence of many irrelevant features. *Artificial Intelligence*, 69(1-2):279–305, 1994.

[26] Kullback S. and Leibler. On information and sufficiency. *Annals of Mathematical Statistics*, (22):79–86, 1951.