Proceedings of the 4th International Symposium on Intelligent Robotic Systems SIRS'96, Lisbon, Portugal, July 22-26, 1996

Memory-based Stochastic Optimization for Automated Tuning of Neural Network's High Level Parameters *

Artur Dubrawski **

The Robotics Institute, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, PA 15213, USA

Abstract. In this paper we describe a new method for automated tuning of high level parameters of supervised learning systems. It uses memory-based learning principles and follows certain ideas of experimental design. The described method allows not only for an efficient search through a decision space, but also for a concurrent validation of the learning algorithm performance on a given data. Potential usefulness of the proposed approach is illustrated with the Fuzzy-ARTMAP neural network application to learning a qualitative positioning of an indoor mobile robot equipped with sonar range sensors. Automatically selected neural network setpoints reach a comparable performance to those achieved by human experts in relatively simple 2D cases. Migration of the proposed method to higher order optimization domains bears a big promise, but requires further research.

1 Introduction

A curse of neural enginnering, ie. the necessity of assigning proper values to neural network's high level parameters (such as a number of hidden layers and hidden units, a learning speed, momentum factors, categorization resolution, and so on) is apparently a serious practical drawback of neurocomputing attempts to tackle various, including robotic, real world problems [7]. Typically, it takes an expert a significant time to achieve a satisfactory performance of a neural system, however the design effort strongly depends on the particular task setup and on the characteristics of a selected network type.

A vast majority of attempts to automate neural networks' high level parameter tuning is based on the evolutionary computation. Due to the space limits we do not include a review of such methods here (for up to date reviews refer to [1, 14]), instead we only express a general opinion that they are usually very expensive in terms of a number of network configurations to try before coming up with a satisfactory one. In contrast, new optimization techniques emerging from experimental design [2] and memory-based learning [11], offer a very attractive way of using both the computational time and the empirical knowledge about the neural network's performance so far.

In this paper we describe a concept and preliminary results obtained with memory-based stochastic optimization [10] used to spot useful and statistically valid settings for high level parameters of the Fuzzy-ARTMAP neural network trained to predict a position of a mobile robot equipped with ultrasonic range sensors. The proposed method of validation and tuning may be actually applied to any kind of a supervised learning algorithm adjusted with numeric gains.

2 Fuzzy-ARTMAP's parameters

Fuzzy-ARTMAP [4] is a modular architecture composed of two Fuzzy-ART [3] networks interconnected with an associative memory, and capable of a stable supervised learning of multidimensional mappings in response to arbitrary sequences of real valued training vectors. The component Fuzzy-ART modules are self-organizing incremental categorization engines. During learning they create efficient discrete representations of arbitrary input data populations.

* This work has been supported by William J. Fulbright Foundation Junior Research Grant to Artur Dubrawski. The author wishes to express his deep gratitude to Andrew W. Moore and Jeff Schneider of the Robotics Institute, CMU, for their manifold help with this paper preparation.

** Email: awd@cs.cmu.edu. The author is on leave from the Institute of Fundamental Technological Research, Polish Academy of Sciences, Warsaw, Poland, (adubr@ippt.gov.pl).

Fuzzy-ART self-organization provides the whole system with a very useful feature of completely autonomous topology development. One does not face difficult choices of numbers of hidden layers and hidden nodes to obtain the best prediction accuracy, when dealing with Adaptive Resonance Theory systems. However, there remains a set of numeric parameters in the model. Particular values of the parameters determine (given the training data) system's eventual topology and performance, thus a proper selection of these values is a key design issue in the Fuzzy-ARTMAP applications. Unfortunately, despite a growing popularity of the ART neural networks and an increasing number of their successful uses (also in robotics: [5, 6, 7, 12]), there are no complete design principles developed so far.

Fuzzy-ART recognition cycle follows a winner-takes-all competitive activation and learning rule. Among the recognition categories, composing the internal representation of the network's knowledge about the input space $\mathbf{w}$ ($0 \leq w_{ij} \leq 1$, i: dimension of the input space, j: number of established categories), the one which seems to be the most similar to the given input pattern $\mathbf{I}$ ($0 \leq I_i \leq 1$) is chosen for further analysis:

$$J = argmax_j \left(\frac{\mid \mathbf{I} \wedge \mathbf{w}_j \mid}{\alpha + \mid \mathbf{w}_j \mid} \right) \tag{1}$$

Where $\mathbf{a} \wedge \mathbf{b} = \mathbf{c} : c_i = \min(a_i, b_i)$ and $\mid \mathbf{a} \mid = \sum_i a_i$. The selected category J is then subject to a match criterion test:

$$\mid \mathbf{I} \wedge \mathbf{w}_J \mid \geq \varrho \cdot \mid \mathbf{I} \mid \tag{2}$$

Then, if the match is satisfactory, Fuzzy-ART iteratively updates its internal representation of the winning recognition category vector (initially all $w_{ij} = 1$):

$$\mathbf{w}_J := \beta \cdot (\mathbf{I} \wedge \mathbf{w}_J) + (1 - \beta) \cdot \mathbf{w}_J \tag{3}$$

If the match test failed, Fuzzy-ART seeks another recognition category likely to pass the test. In absence of such categories, a new one is created to represent the unmatched input vector.

In the equations above α, β and ϱ are high level parameters, determining the performance of the Fuzzy-ART recognition and category learning processes. All of them take real values in a range from 0 to 1. The vigilance parameter ϱ controls the fineness of the established categories and generalization properties of the network. Smaller values of ϱ result in coarser classes covering larger regions of the input space, whereas the larger ϱ is the more specific and finer internal representation become, approaching a lookup table at the limit of $\varrho = 1$. Very small values of the choice parameter α force the network to pick larger categories in case when more than one of them is close to the input pattern, ie. the network prefers to make conservative choices. Parameter β controls the adaptation speed. If $\beta = 1$ the networks learns in one shot, if $\beta < 1$ a slow learning takes place (which is usually a reasonable choice for dealing with noisy input data). A typical setup of the Fuzzy-ART module assumes fast learning of newly committed categories and slow adaptation of already existing ones.

In a fully configured Fuzzy-ARTMAP system there are two sets of three Fuzzy-ART parameters, one for each of the modules. They are indexed α^a, β^a, ϱ^a and α^b, β^b, ϱ^b respectively. In certain applications one may wish to use different settings for learning and recall modes of operation, by introducing α_r^a, ϱ_r^a and α_r^b, ϱ_r^b when learning is disabled.

Fuzzy-ARTMAP system employs an associative map $\mathbf{w}^{ab}$ ($0 \leq w_{jk} \leq 1$, j: number of established input categories, k: number of established output categories) to learn relationships between internal recognition categories of the two Fuzzy-ART modules. In a recall mode, an input $\mathbf{I}^a$ selects a category coded by $\mathbf{w}_J^a$, what leads to activation of an output category K, such that:

$$K = argmax_k (w_{Jk}^{ab}) \tag{4}$$

and $\mathbf{w}_K^b$ vector codes the generated output prediction. If that prediction does not match a desired output, or if the following criterion is not satisfied:

$$w_{JK}^{ab} \geq \varrho^{ab} \tag{5}$$

then an iterative search for satisfactory matches continues until a success or a forced commission of a new input category. The associative map representation is updated according to the following formulas (with all uncommitted w_{jk} initially set to 1):

$$\begin{aligned} \mathbf{w}_{Jk}^{ab} &:= (1 - \beta^{ab}) \cdot \mathbf{w}_{Jk}^{ab} \quad \text{for } k \neq K; \\ \mathbf{w}_{JK}^{ab} &:= (1 - \beta^{ab}) \cdot \mathbf{w}_{JK}^{ab} + \beta^{ab} \end{aligned} \tag{6}$$

Again, ϱ^{ab} and β^{ab} are real numbers taken from the range 0 to 1, and ϱ^{ab} could have a special setting for recall purposes ϱ_r^{ab}.

As we see from a simplified description of the Fuzzy-ARTMAP operation, there are up to 13 numeric high level parameters to tune, which is obviously not an easy task in general. Nevertheless some practical experience gathered so far allows designers to reduce that number down to 2 or 3 (ϱ^a, β^a and ϱ^b) in particularly nice cases. But in fact, especially attacking a real world problem, one would rather keep the number of free high level parameters as large as possible given design time constraints. In the following section we briefly describe a computationally efficient and potentially very useful approach to automated tuning of the Fuzzy-ARTMAP high level parameters.

3 Memory-based stochastic optimization

Performance of the Fuzzy-ARTMAP system is deterministic given the data, ie. it would reveal always the same results over a given testing set, after being taught using a given training set with the same high level parameters setup. Unfortunately, as in the case of any other supervised learning algorithm, validation based on a single fold of the available data (so called train-and-test or hold-out method [13]) gives a cheap, but usually biased estimate of the system performance. A much more relevant approximation can be obtained with cross-validation or bootstraping, but these techniques require several repetitive train-and-test cycles, and thus are relatively expensive. The larger the factor K of the K-fold cross-validation, the better the learner's performance estimate, and the more train-and-test cycles need to be executed at a single setpoint of the high level parameter space. We argue that for most cases it is more efficient to perform only a single fold of the cross-validation or bootstraping at a given point, and use the remaining time for sampling $K-1$ times among some other places of the domain. If for each experiment the available data is randomly split into the training and the testing subsets of the sizes defined by the folding factor K, we cannot expect a deterministic behavior of the learner anymore. However, the performance estimates obtained that way can still be compared against each other, as noisy values calculated upon random folds of the same population. Moreover, if we were sampling at a selected point for a sufficient number of times, we would eventually obtain a confident estimate of the system performance at that point, which would be as good (or better) as the one computed with respective cross-validation.

With the above described stochastic approach to validation we have managed to turn a deterministic machine learning algorithm into a random process relevant to the memory based optimization.

Memory based stochastic optimization [10] is a new technique combining memory-based learning [11] with experimental design methods [2]. It differs form a conventional numerical optimization because it accepts noisy samples, operates on non-linear or just locally linear approximations of the objective function surface, makes use of uncertainty of the maintained objective function model, and attempts to perform a relatively small number of deliberatively selected optimization steps. These features make memory-based stochastic optimization exceptionally suitable for a class of tasks in which gathering empirical data about the optimized process outcomes is costly, risky, time consuming, and if the gathered data is noisy.

Memory-based learning builds a locally weighted regression model of the observed process input-output behavior, based upon all observations made so far. If the observations are noisy, the local regression fit provides a filtered estimate of the expected output value. A query at a selected point of the model reveals not only the output expectation, but also Bayesian estimates of its confidence [10]. Experimental design methods use that information to suggest input space locations to try, in order to maximize the expected output and/or efficiently enrich the model. Results of the execution of the experiments are then used for the model update. For the purpose of this paper we consider three of the experimental design heuristics, briefly described below.

Response surface methods (RSM) are statistical techniques traditionally used in experimental design [2]. At the beginning of a search for an optimum, a region of interest is determined, and experiments are made at points indicated by a low-order polynomial regression as locally the best for identyfying the underlying objective function's properties. As soon as the local gradient's estimate gets accurate enough, a statistician moves the region of interest towards promising areas of the problem space, and with a quadratic regression detects local optima, resolves saddles and ridges, and tracks the local gradient accordingly. Unfortunately a conventional RSM requires human intervention. Moreover it makes no use of old data, ie. as soon as the region of interest is changed, the method makes no reference to the previously explored areas. These are fundamental motivations for the application of a memory-based learning [11] to retain the data gathered during exploration, and to build a global nonlinear plant model with them. The model, containing the information on the objective function and on its confidence, significantly improves efficiency of the RSM search decisions and makes an algorithmization of the technique possible. It is called AutoRSM [10].

Response surface optimization is designed to make rather cautious changes to operating conditions of the analyzed process. Quite an opposite idea led to the IEMAX algorithm [10], which is an extension of the interval estimation method [9] to continous domains. IEMAX suggests experiment locations with the principle of optimism. If we think of a locally weighted regression model of the process outputs as a surface in the process input space, then - given requested confidence - we can also think of surfaces representing upper and lower limits of the output confidence intervals. IEMAX tries to spot a maximum of the upper confidence limit surface, which then becomes the next experiment suggestion.

The third of the selected experimental design methods, called OPTEX [8], makes its suggestions at the location of the largest expected improvement over the current model's maximum y^* [8]:

$$\mathbf{x}_{sugg} = \max_{\forall \mathbf{x}} \left(\int_{y_{min}}^{y_{max}} max(0, y - y^*) \cdot p(y \mid \mathbf{x}) dy \right) \tag{7}$$

where $\mathbf{x}$ is the input query vector, y, y_{min}, y_{max} are the output and its limits, and $p(y \mid \mathbf{x})$ is a probablility density function of y at a given query point $\mathbf{x}$.

4 Learning task setup

The particular robotic task we consider here is a mobile robot qualitative localization in a setup described in more detail in [12]. A robotic vehicle, equipped with ultrasonic range sensors, is placed somewhere in proximity of a selected object of the scene (a doorway in particular). A Fuzzy-ARTMAP neural network is trained in a supervised mode to predict sectors of location in a doorway neighborhood region, given raw readouts of 9 ultrasonic range sensors mounted on the robot. The data collected on a real vehicle[1] cover

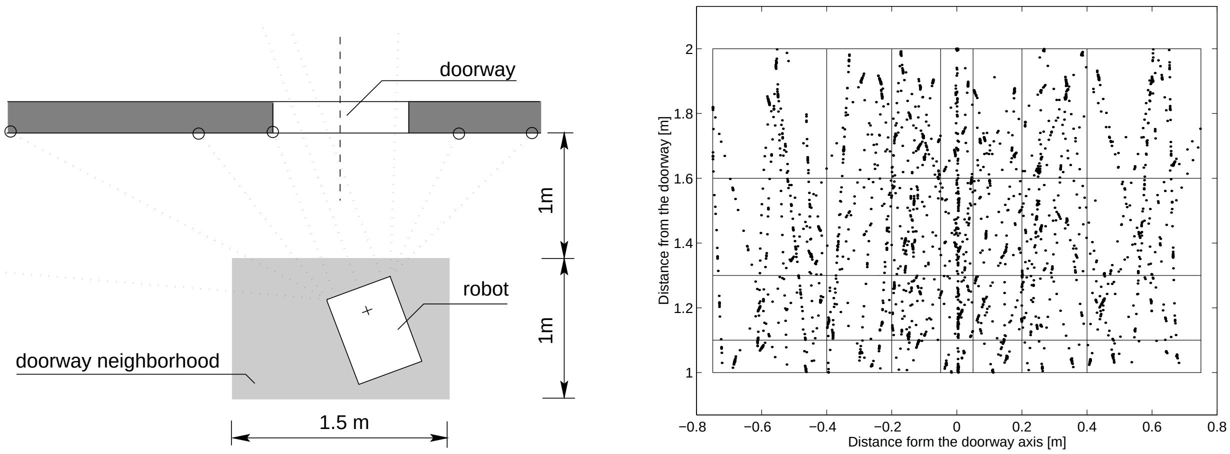

Fig. 1.: Task setup, data layout and position sectors in the doorway neighborhood region.

a rectangular area 1.5 meter width and 1 meter deep, located symmetrically on the doorway axis, 1 meter away from the doorway center (Fig. 1). The region is preclassified into 28 non-overlaping rectangular sectors of cartesian 2D position. In this setup, the Fuzzy-ARTMAP architecture reduces to a single Fuzzy-ART module and an associative map that links input recognition categories with predefined output classes of the location setors. The number of the high level parameters reduces accordingly to 8, but in fact we limit the degrees of freedom even further in some of the experiments discussed below.

At each validation cycle learning is performed off-line with a randomly selected part of the available data. An ideal predictor would give correct position estimates when supplied with the remaining data samples.

[1] Data collected by Janusz Racz on *Robuter* at LIFIA IMAG, Grenoble, France.

Quality of prediction is thus measured as a cummulative relative misclassification error over the testing data set. The prediction accuracy depends of course on the representativeness of the collected data, on the amount of noise in it, and on the way it is splitted into training and testing parts. Otherwise it depends mainly on the particular choice of the high level parameters' settings.

In the research described in this paper we additionally constrain the size of memory occupied by a neural network representation. It has an obvious practical justification and extends the work [12]. The objective function reflects the prediction accuracy and the network's representation size:

$$f(\epsilon, M) = \kappa \cdot (1-\epsilon) + (1-\kappa) \cdot [1 - (M/M_{max})^2] \tag{8}$$

where $0 \leq \epsilon \leq 1$ is a misclassification error obtained with a single run of the stochastic five-fold validation over a population of 5177 data samples, M is a size of memory taken by the Fuzzy-ARTMAP representation, M_{max} is a memory limit (65536 bytes in the presented experiments), and finally the weighting factor $\kappa = 0.95$.

Note that f is a function of two random variables, and we expect it to reveal a certain variance due to a stochastic way of splitting the original data set into the training and testing parts. The optimization task is to determine which region of the high level parameters space has the largest expected value of the objective function, provided that the expectation is confident enough.

5 Experiments

A purpose of the presented experiments is to verify an adequacy of the memory-based stochastic optimization for tuning high level parameters of a learning algorithm, and figure out possible profits from its application to a realistic robotic task.

First of all we would like to see whether the method is capable of spotting useful setups in an acceptable number of validation steps. In order to make the heuristics comprehensible we constrain the optimization task to a two-dimensional case by choosing ρ^a and β^a as decision variables and setting the remaining six parameters to reasonable values suggested by a human expert ($\alpha^a = \alpha_r^a = 0.01$, $\varrho^{ab} = \varrho_r^{ab} = \beta^{ab} = 1$, $\varrho_r^a = 0.9\varrho^a$). It is sensible to set the number of experimental steps to 200, which is equivalent to 40 complete five-fold cross-validation tests. If we chose cross-validation, we would gather information at merely 40 points of the decision space with the same computing effort. With 200 noisy samples we expect to obtain decent, if not perfect, regression models of the learner's performance, spot promising regions of the decission space, and achieve a reasonable confidence about the predicted optima. At each validation step the network executes 3 epochs of training, then memory usage is calculated, and the predictive accuracy is determined with use of a testing data set.

In Fig. 2,3 and 4 a development of regression models is shown. AutoRSM (Fig. 2) happen to fall into a region likely to contain a maximum of the underlying model at the very beginning of the search. Due to cautious behavior principles, it never left that area. Instead, it kept intensively sampling in the region of interest. In less than 80 steps however (see Fig. 5), AutoRSM achieved a very high confidence of the estimated optimal value of the objective function model. After 150 steps, at the suggested optimum location ($\rho^a = 0.9056$ and $\beta^a = 0.9062$) the expected value of the objective function was 0.8253, and 95th %-ile confidence limits equal 0.8215 (low) and 0.8291 (high). The classical 5-fold cross-validation revealed at the same point an almost identical objective function estimate (0.8242), average predictive error of 17.58% and average memory consumption of 55872 bytes.

Both IEMAX (Fig. 3) and OPTEX (Fig. 4) behave much more vigorously than AutoRSM. In just 20 steps they begin to discover the most important feature of the modeled system, ie. a steep slope at high values of ρ^a caused by the imposed memory usage limit. In about 100 steps both methods decided to pay their attention to the region along that edge, correctly expecting to spot a maximum of the objective function over there. After approximately 120 steps IEMAX became significantly confident about predicted maxima. The expected value of the optimum was however surprisingly low (0.8070 at $\rho^a = 0.8300$ and $\beta^a = 0.8540$, with the 95-percentile confidence limits 0.7938 (low) and 0.8175 (high)). 5-fold cross-validation returned a larger value of the objective estimate at the same point (0.8240, av. pred. error 17.56%, av. memory usage 45523 bytes). We suspect that the difference between these estimates arises from a locally too general regression model, which may lead to underfitting of the true distribution in the regions, where a density of samples is relatively high. Nonetheless that hypothesis requires further investigation. We observed a smaller discrepancy with OPTEX, which in about 170 validation steps began to stabilize at $\rho^a = 0.8450$ and $\beta^a = 0.7750$ predicting objective

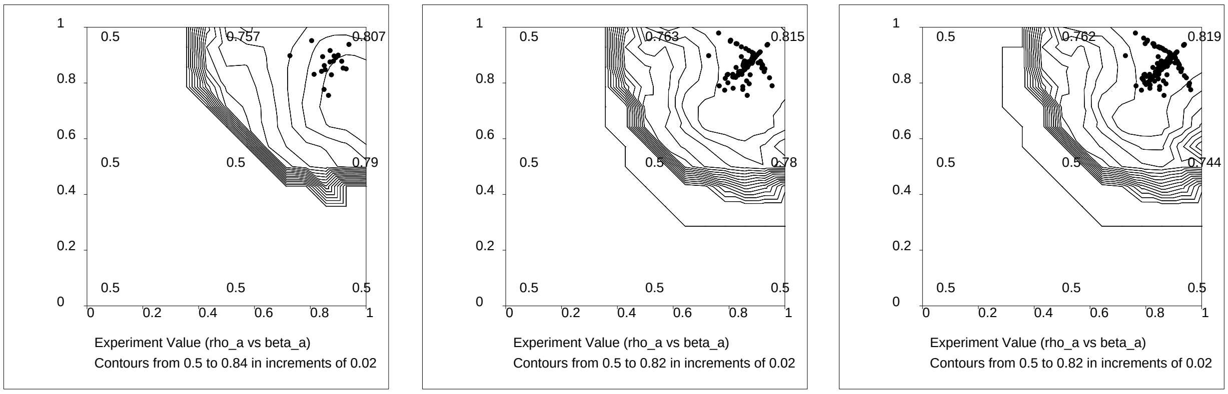

Fig. 2.: Evolution of a regression model in course of a 2-dimensional parameter optimization with AutoRSM: contours of equal objective function values after 20, 100 and 180 validation steps plotted in coordinates ϱ^a (horizontal axis) and β^a (vertical). Black dots depict locations of the evaluated Fuzzy-ARTMAP setups.

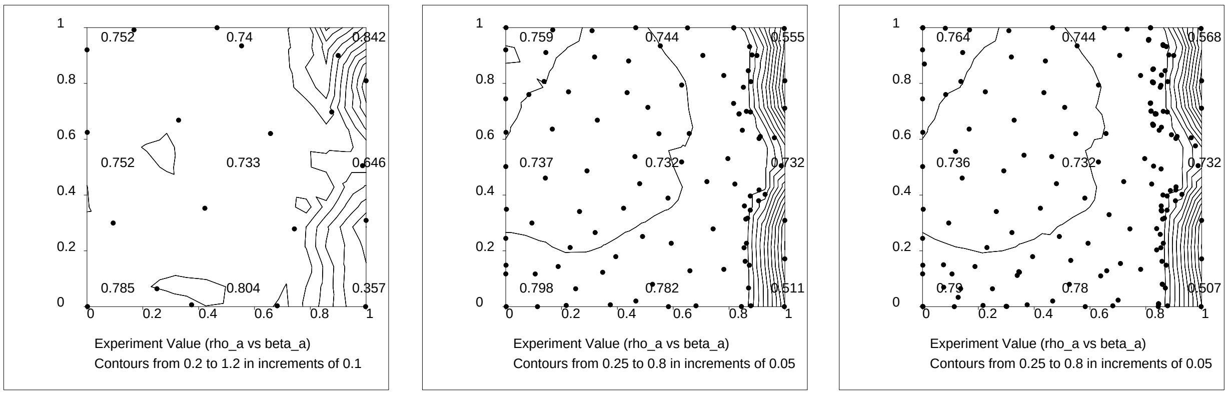

Fig. 3.: Evolution of a regression model with IEMAX: contours of equal objective function values after 20, 100 and 180 validation steps (2-dimensional case).

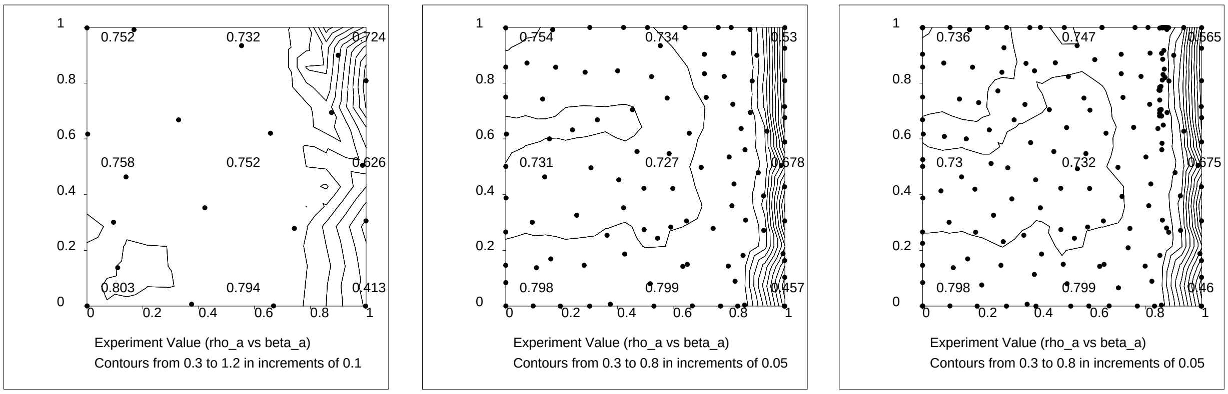

Fig. 4.: Evolution of a regression model with OPTEX: contours of equal objective function values after 20, 100 and 180 validation steps (2-dimensional case).

value of 0.8160 with a reasonable confidence (0.8015 low, 0.8313 high). 5-fold cross-validation estimated 0.8242 (av. pred. error 17.57%, memory usage 47315 bytes), what fell within the OPTEX 95%-ile interval.

It is interesting to compare the performance of automated tuning of the high level parameters against the outcomes provided by human experts. Work [12] presents the results of manual tuning of the Fuzzy-ARTMAP system in the same as ours robotic application. The best predictive accuracy reported there was 16% at $\varrho^a = 0.89$ and $\beta^a = 0.3$, calculated with the hold-out method. These results were achieved in 5 epochs of training, after an unspecified number of manual optimization experiments. Our setup is just slightly different (we allow 3 epochs of training, use possibly less biased validation methods, additionally restrict memory usage, and, worst of all, perform tests over a slightly different data population). Applying the 5-fold cross-validation in our setup at the same setpoint as above we obtained an expected objective function value as high as 0.8341, predictive error of 16.6% and average memory usage 56600 bytes. These results slightly, but clearly, outperformed those obtained with automated tuning.

Despite that, the true power of automated tuning should become clear in a more than just 2-dimensional decision space. Among the tested experimental design heuristics, so far only OPTEX demonstrated capability of achieving affordaable confidence about the predicted optima in 8-dimensional space within 200 validation cycles (Fig. 6, right part). At the 123th step the reported optimal objective function value was 0.8538 (at $\varrho^a = 0.357$, $\beta^a = 0.914$, $\alpha^a = 0.9997$, $\varrho^a_r = 0.3506\varrho^a$, $\alpha^a_r = 0.01 = 0.97$, $\varrho^{ab} = 0.997$, $\beta^{ab} = 0.6367$, $\varrho^{ab}_r = 0.9383\varrho^{ab}$). The 5-fold cross-validation prediction at the same point scored 0.8553 with average error 14.47% and memory usage of 48379 bytes, which was the best result obtained so far.

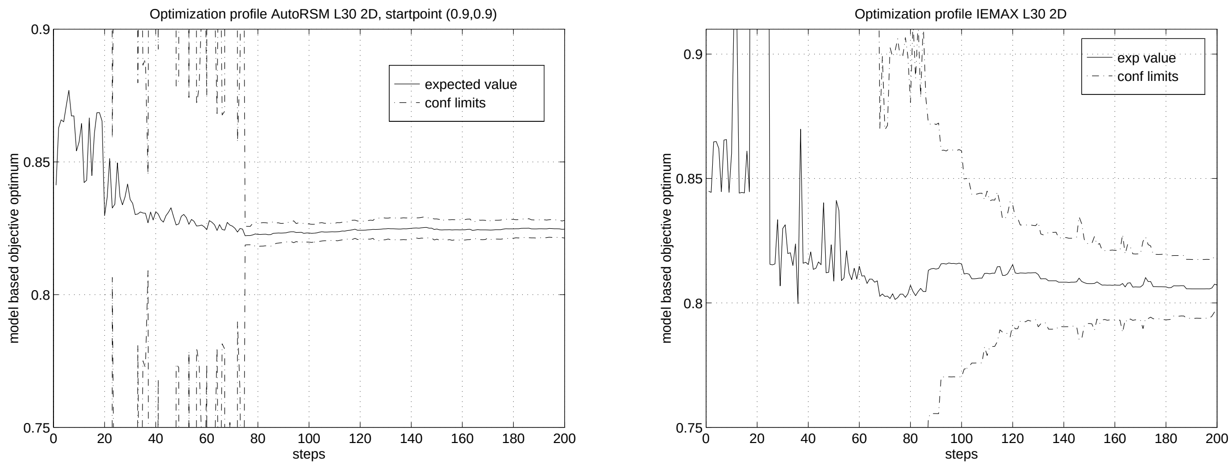

Fig. 5.: Profiles of a 2D optimization (decision variables ρ^a, β^a): model based estimate of the objective function optimum and its 95th %-ile confidence limits vs. number of validation steps for AutoRSM and IEMAX.

6 Conclusion

In this paper we introduced a new method for automated tuning of high level parameters of supervised learning algorithms, combined with an experimental validation of these algorithms performance. We verified the method on a fairly well studied practical robotic task with use of realistic, noisy and nonuniformly distributed training data.

The results obtained in a reasonably few experimental steps are almost as good as those achieved by human experts in a simplified, two-dimensional optimization problem domain. A preliminary attempt in an 8-dimensional space revealed an immediate improvement over a human-generated outcome.

We expect to easily outperform the alternative - evolutionary motivated techniques - in case of optimization in low dimension spaces (nb. the most common among the supervised learning algorithms), however high-dimensional decision problems (as the Fuzzy-ARTMAP tuning for example) doubtlessly need further comparative studies.

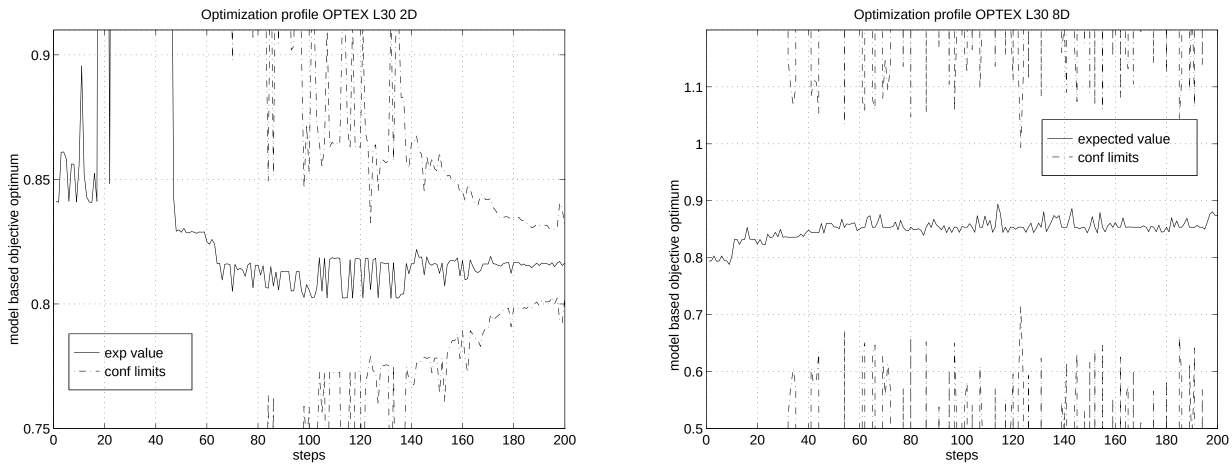

Fig. 6.: OPTEX optimization profiles: model based estimate of the objective function maximum and its 95th %-ile confidence limits vs. number of validation steps obtained with 2D and 8D optimization.

Unfortunately, the current implementation of the memory-based stochastic optimization is rather slow at high dimensional tasks. It is however not clear by now how much speedup can be gained by simple code optimization. Instead, we consider research towards a combined experimental design heuristics, revealing optimistic behavior at the initial stage of optimization in order to roughly spot regions of interest, and a conservative and detailed exploration of these regions at the following stages. Another promising way is a development of multiresolution regression models and relevant experimental design heuristics.

References

1. Balakrishnan K., Honavar V. Evolutionary Design of Neural Architectures - A Preliminary Taxonomy and Guide to Literature. *Technical Reprort CS TR 95-01*, Iowa State University, 1995.
2. Box G.E.P., Draper N.R. *Empirical Model-Building and Response Surfaces*, Wiley 1987.
3. Carpenter G.A., Grossberg S., Rosen D. Fuzzy ART: Fast Stable Learning of Analog Patterns by an Adaptive Resonance System. *Neural Networks*, 759-771, Vol.4, 1991.
4. Carpenter G.A., Grossberg S., Markuzon N., Reynolds J.H., Rosen D.B. Fuzzy ARTMAP: A Neural Network Architecture for Incremental Supervised Learning of Analog Multidimensional Maps. *IEEE Trans. on Neural Networks*, 689-713, 3(5), 1992.
5. Dubrawski A., Crowley J.L. Self-Supervised System for Reactive Navigation. *1994 IEEE International Conference on Robotics and Automation*, San Diego, USA, May 1994.
6. Dubrawski A., Reignier P. Learning to Categorize Perceptual Space of a Mobile Robot Using Fuzzy-ART Neural Network. *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems IROS'94*, Munich, Germany, September 1994.
7. Dubrawski A. Neurocomputing for mobile robot navigation - successful and promising attempts. (unpublished lecture notes) *3rd Int. Symp. on Intelligent Robotic Systems SIRS'95*, Pisa, Italy, July 1995.
8. Dubrawski A., Moore A.W., Schneider J. Memory-Based Stochastic Validation and Tuning of Supervised Learning Algorithms. *Technical Report CMU-RI-TR-96-25*, The Robotics Institute, Carnegie Mellon University, 1996.
9. Kaelbling L.P. Learning in Embedded Systems. *PhD Thesis. Technical Report TR-90-04*, Department of Computer Science, Stanford University, 1990.
10. Moore A.W., Schneider J. Memory-based Stochastic Optimization. *Advances in Neural Information Processing Systems 8*, MIT Press, 1995 (to appear).
11. Moore A.W., Atkeson C.G., Schaal S. Memory-based Learning for Control. *Technical Report CMU-RI-TR-95-18*, The Robotics Institute, Carnegie Mellon University, 1995.
12. Racz J., Dubrawski A. Artificial neural network for mobile robot topological localization. *Robotics and Autonomous Systems*, 16(1), 73-80, 1995.
13. Weiss S.M., Kulikowski C.A. *Computer Systems That Learn*, Morgan Kaufman Publishers, 1991.
14. Yao X. A Review of Evolutionary Artificial Neural Networks. *Int. J. of Intelligent Systems*, 8(4), 539-567, 1993.

This article was processed using the TEX macro package with SIRS96 style