Artur Dubrawski
IPPT PAN

AUTOMATED TUNING OF A NEURAL NETWORK FOR LOCALIZATION OF A MOBILE ROBOT

Abstract: In this paper we present a method of automated tuning of high level parameters of neural networks. It uses memory-based learning principles, follows ideas of experimental design and stochastic optimization, and uses an innovative approach to resampling called stochastic validation. 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 ultrasonic range sensors. Automatically selected setpoints allow the system to reach a comparable performance to that achieved by human experts in 2-D parameter optimization cases, and lead to a better performance in higher dimensions.

1 INTRODUCTION

One of the main difficulties in practical applications of neural networks to real world problems is 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, etc.). The eventual design effort depends on the particular task setup and on the characteristics of a selected network type, but anyway it usually takes an expert a significant tweaking time to achieve a satisfactory performance of a neural system. New optimization techniques emerging from experimental design [2], stochastic approximation [5], and memory-based learning [8], offer a very attractive way of using both the computation time and the empirical knowledge about the system's performance so far to quickly and purposefully spot interesting regions of the decision space.

In this paper we describe a concept and some results obtained with memory-based stochastic optimization applied to experimentally select 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 task of the presented method is to automatically spot reasonable values of the neural network's parameters with as small computational effort as possible. It is designed to select the experiments purposively and to perform the most informative ones. It also takes care on making an efficient use of the experimental data collected so far. To do so it builds probabilistic models of relationships between the values of the high level parameters and the network's performance. Those models are then used to predict optimal settings and to suggest further experiments.

The proposed method of validation and tuning may be actually applied to any kind of a supervised learning algorithm, which is controlled with some continuous numeric high level parameters. In case of discrete parameters a technique of *racing* would be more suitable [6].

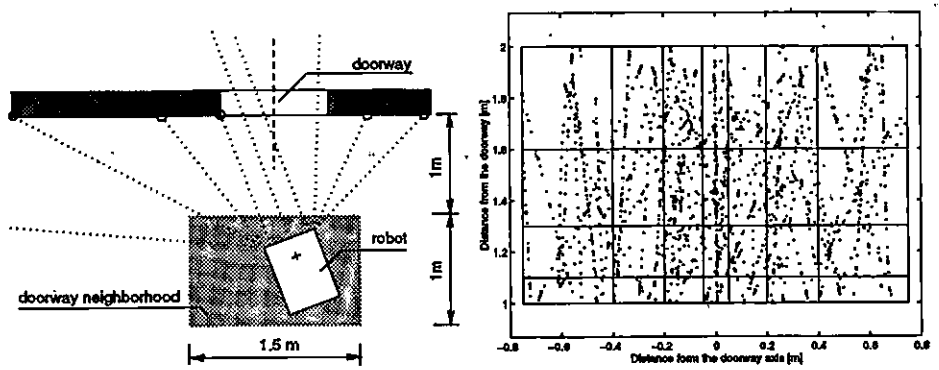

Figure 1: Localization task setup, data scatter and position sectors.

2 NEURAL NETWORK FOR A MOBILE ROBOT LOCALIZATION

The particular robotic task we consider in this paper is a mobile robot qualitative localization in a setup described in detail in [9]. A robotic vehicle (Robosoft *Robuter*), equipped with ultrasonic range sensors, is placed somewhere in proximity of a selected object of the scene (a doorway in particular). The task for a trained neural network is to predict a relative position of the robot, given immediate readouts of 9 ultrasonic range sensors, assuming that the robot is placed somewhere in a doorway neighborhood region.

The data for training has been collected on a real vehicle and covers a rectangular area 1.5 meter width and 1 meter deep, located symmetrically along the doorway axis, 1 meter away from the doorway center (Fig. 1). That region has been preclassified into 28 non-overlaping rectangular sectors of a cartesian 2-D position. The data collection has a form of a set of input–output pairs which associate sensory readouts with location sectors.

An essential evaluation criteria for a neural network in this setup is prediction accuracy. To assess it we perform validation cycles. At each cycle the network is trained in a supervised mode using a randomly selected part of the available data. An ideal predictor would give correct position estimates when supplied with the remaining data which has not been seen by the system during training. Prediction accuracy is thus measured as a cummulative misclassification error calculated over the testing part of the data set. The accuracy depends of course on the relevance of the collected data, on the amount of noise in it, and on the way it is being split into training and testing parts. Otherwise, the prediction accuracy depends mainly on the particular choice of the high level parameter settings, which control a behavior of the machine learning algorithm in use.

In the research described in this paper we additionally constrain the size of memory occupied by a neural network knowledge representation. It has an obvious practical justification and extends the original work [9]. The objective function reflects the prediction accuracy and the network's representation size together:

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

where $0 \leq \epsilon \leq 1$ is a misclassification error obtained during a single validation cycle over a population of total 5177 data samples, M is a size of memory taken by the neural network representation, M_{max} is a memory limit (65536 bytes in the presented experiments), and

finally the weighting factor κ equals 0.95. Note that f is a function of two random variables (ϵ and M), 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.

The neural network of our choice, Fuzzy-ARTMAP, has a modular architecture composed of two Fuzzy-ART networks interconnected with an associative memory. It is 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 input data populations. Fuzzy-ART self-organization provides the whole system with an 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. Their particular values determine, given the training data, the system's eventual topology and performance. So, a proper selection of these values is a key design issue in the Fuzzy-ARTMAP applications. Interestingly, despite the established popularity of the ART networks and an increasing number of their successful uses (also in robotics [1]), there are no complete design principles developed so far.

Due to the space limits we have to skip a detailed Fuzzy-ARTMAP operation description here. An interested reader may refer to the fundamental literature [3]. For this paper purposes it is enough to say that the complete Fuzzy-ARTMAP may be controlled by as many as 13 numeric parameters, but such a level of sophistication is rarely considered in practice. In the experimental setup we discuss in this paper we have an access to up to 8 high level parameters, namely to the vigilance factors of the input network (ϱ^a in a training phase and ϱ^a_τ in a recognition phase) and of the map field (ϱ^{ab} and ϱ^{ab}_τ), to the input network choice parameters (α^a, α^a_τ), and to the learning speeds (β^a and β^{ab}). For comparison reasons in some of the experiments we restrict the number of the tuned parameters even further (down to two: ϱ^a, β^a).

3 MEMORY-BASED STOCHASTIC VALIDATION AND TUNING

Performance of the Fuzzy-ARTMAP system is deterministic given the data. That is, it would reveal always the same results over a given testing set, after being taught using a given training set and the same training samples sequence, 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) gives an inexpensive, but usually a high variance estimate of the performance. A more relevant approximation can be obtained with cross-validation, but that technique requires several repetitive train-and-test cycles, and thus it is 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 parameters space. We argue that for most cases it is computationally more efficient to perform only a few folds of the K-fold cross-validation at a given point, and use the remaining time for sampling among some other places of the domain.

If for each experiment the available data was *randomly* split into the training and the testing subsets of the sizes defined by the constant folding factor K, we cannot expect a

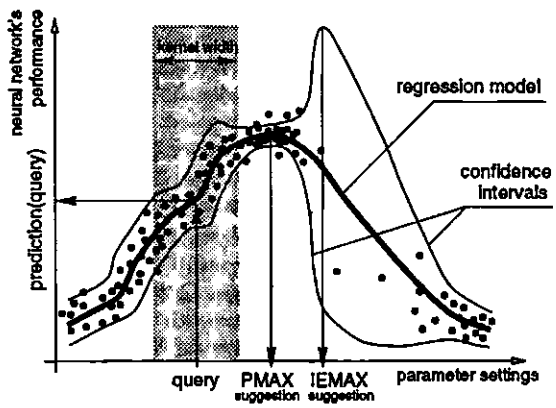

Figure 2: Memory-based learning by locally-weighted linear regression for a case of one-dimensional input (horizontal axis) and one-dimensional output (vertical axis). Black dots correspond to the experimental data, thick line is a locally weighted linear regression model of the observed process, thin lines depict model's confidence intervals. Kernel width determines how local is the regression, ie. how important are the datapoints located far from the given input query point vs. those located nearby, in making the prediction.

deterministic behavior of the learner anymore. That is, the learner's performance estimates calculated at a selected setpoint of its high level parameters will, except very special cases, be different for different pairs of training and testing sets drawn randomly from the same source data. These estimates however can still be compared against each other as noisy values computed upon random folds of the same data population. Moreover, if we were sampling at a selected setpoint of the high level parameters for a sufficient number of times, we would eventually obtain a confident estimate of the system performance at that point, which would be at least as relevant as the one computed with the respective deterministic K-fold cross-validation.

With the above described resampling scheme, which we call *K-fold stochastic validation*, we trade a slight increase of variance of the learning algorithm performance evaluations for substantial savings in computational effort. Stochastic validation turns an originally deterministic behavior of a machine learning algorithm into a random process suitable for the memory-based optimization.

Memory based stochastic optimization [7] is a new technique that combines memory-based learning [8] with experimental design methods [2]. It differs from 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 (Fig. 2). Experimental design methods use that information to suggest input space locations to try, in order to maximize the expected output and/or enrich the model. Results of the execution of the experiments are then used for the model update.

If we think of a memory-based 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* algorithm [7] tries to spot a maximum of the upper confidence limit, which then becomes the next experiment suggestion. Usually outcomes of the experiments suggested in that way are lower than the predictions given by the upper confidence limits. But, as a result of updating the memory-based regression model with the new outcome, the upper confidence interval surface is being nailed down towards the actual outcome, and the performance model surface is being reshaped to better represent the actual learner's performance. Then IEMAX suggests a new experiment at the location of a maximum of the updated upper confidence interval and continues aggresive exploration throughout the entire search space in hope of finding the global optimum.

Another experimental design method, called *OPTEX* [7], makes its suggestions at the location of the largest expected improvement over the current model's maximum. The improvement expectation at a given input query point is estimated by the way of integration of the upper tail of a conditional probability density function of the model prediction given the query. The integration begins from the pdf value related to the current model's maximum. OPTEX has a potential of finding global optima in a sensible manner, but behaves slightly less spontaneously than IEMAX and needs more computation.

The last of the search heuristics described in this paper, called *PMAX*, does not have a direct application to stochastic optimization. Its suggestions correspond to the maxima of the memory-based model of the learner's performance. PMAX's recommendations can be seen as predictions of the best performance spotted so far, and we use them to evaluate progress during optimization with IEMAX or OPTEX.

By using IEMAX or OPTEX over locally weighted linear regression models of the learner's performance, estimated with stochastic validation, it is possible to quickly determine promising regions of the high level parameters space. During tuning, the amount of time spent while evaluating regions of low expectations is being reduced. The experiments are being performed more often along promising areas of the space, providing us in return with more confident and more accurate evaluations right there, where they are needed.

4 EXPERIMENTS

A purpose of the presented experiments was to verify the adequacy of the memory-based stochastic optimization for tuning high level parameters of a learning algorithm, and to 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

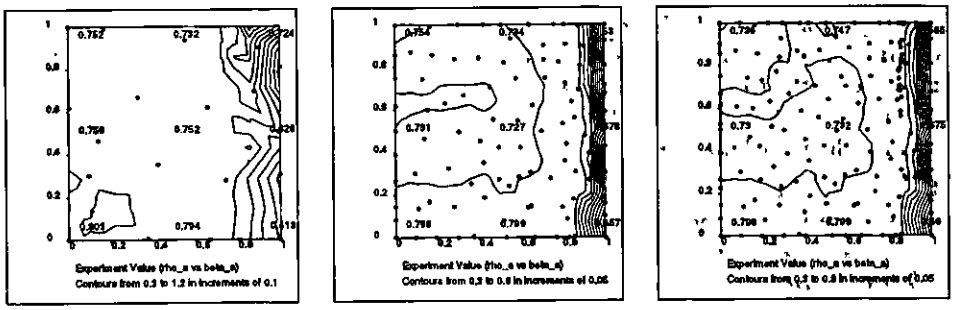

Figure 3: Evolution of regression models obtained with OPTEX: contours of equal objective function values after 20, 100 and 180 validation steps (2-D case) plotted in coordinates ϱ^a (horizontal axis) and β^a (vertical). Black dots depict locations of the evaluated Fuzzy-ARTMAP setups. Contours obtained with IEMAX were similar.

5-fold cross-validation tests. If we chose full five-fold cross-validation, we would gather information at merely 40 points of the decision space, whereas by taking into account only one training-testing data fold per point we would be able to visit 200 points with a similar computing effort. With 200 noisy samples we expect to obtain decent regression models of the learner's performance, to spot promising regions of the decision space, and to 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. 3 a development of the regression models is shown. Both IEMAX and OPTEX in just 20 steps 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 observed a smaller discrepancy with OPTEX though, which in about 170 validation steps began to stabilize at $\rho^a = 0.8450$ and $\beta^a = 0.7750$, predicting objective 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 [9] presents the results of manual tuning of the Fuzzy-ARTMAP system in the same as our 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. Those results were achieved in 5 epochs of training, after an unspecified but large number of manual optimization experiments. Our setup is just slightly different (we allow 3 epochs of training, use possibly more relevant validation methods, restrict memory usage and perform tests over a slightly different data population). By applying the 5-fold cross-validation in our setup at the same setpoint as above, we obtained the objective function value as high as 0.8341, predictive error of 16.6% and

	Optimizer:	IEMAX 2-D	OPTEX 2-D	OPTEX 8-D	Human
PMAX predictions	predicted optimal value	0.8070	0.8160	**0.8538**	n.a.
	confidence interval half-width	0.0170	**0.0149**	0.2676	n.a.
5-fold CV scores	score at the predicted optimum	0.8240	0.8240	**0.8553**	0.8341
	predictive accuracy	17.56%	17.57%	**14.47%**	16.6%
	average memory usage (bytes)	**45523**	47315	48379	56600

Table 1: Summary of the results revealed by IEMAX and OPTEX working on 2-D optimization task, OPTEX tuning eight high level parameters, and the results obtained using suggestions provided in [9] by a human expert.

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-D decision space. Among the tested experimental design heuristics, so far only OPTEX demonstrated capability of achieving affordable confidence about the predicted optima in eight-dimensional space within 200 validation cycles. At the 123rd step the reported optimal objective function value was 0.8538 (at $\varrho^a = 0.357$, $\beta^a = 0.914$, $\alpha^a = 0.9997$, $\varrho_r^a = 0.3506\varrho^a$, $\alpha_r^a = 0.01 = 0.97$, $\varrho^{ab} = 0.997$, $\beta^{ab} = 0.6367$, $\varrho_r^{ab} = 0.9383\varrho^{ab}$). The 5-fold cross-validation prediction at the same point scored 0.8553 with the average error 14.47% and memory usage of 48379 bytes, which was the best overall result obtained so far. Table 1 contains a summary of the results revealed by IEMAX and OPTEX when working in two dimensions, OPTEX when optimizing the 8-D case, and those obtained by a human expert (after [9]).

5 CONCLUSION

In this paper we presented a method for automated tuning of high level parameters of supervised learning algorithms, combined with an experimental validation of their performance. We verified the method on a neural network applied to a fairly well studied practical robotic task, with use of realistic, noisy and nonuniformly distributed training data. In fact, the method may be directly applied to any kind of a supervised learning algorithm, which is controlled with continuous numeric high level parameters.

The results obtained in a reasonably few tuning steps are almost as good as those achieved during an extensive manual experimentation performed by human experts in a two-dimensional optimization case. An attempt in eight dimensions revealed an immediate improvement over a human-generated outcome.

We are directing our future research towards using adaptive kernel width in locally weighted regression models, so that model's resolution might vary along the search space from fine around regions of high expentance and high data density, to coarse elsewhere. Such an extension should make the confidence intervals adaptation smoother, and thus allow IEMAX and OPTEX to avoid low outcome regions of the high level parameters space in the earlier stages of optimization than they do in the current setup.

Another interesting issue is related to multicriterial optimization. It is possible to build regression models for each of the decision attributes separately, instead of maintaining a single model for an objective function of multiple attributes. Separate models would be

independent on the particular form of the objective function.

One of practical questions in tuning the learning algorithms is which of the high level parameters (or their combinations) are vital, and which of them do not influence the learner's performance very much. With an experimental knowledge collected in the memory-based models we have a potential to answer that question.

The presented method of validation and tuning of learning algorithms relies on the proposed stochastic resampling scheme. In this paper we limited our discussion to the case in which at each optimization step only one training-testing data fold was taken into account. In fact K-fold stochastic validation does not impose such a limit. It is legal to take into consideration outcomes obtained with more than one random data split at each optimization step. It would be interesting to see what is the optimal amount of resampling, given optimization time constraints. An intrigued reader will find some insights on that matter in [4].

The future research issues will be examined using the presented mobile robot localization setup and on a cellular neural network application to a mobile robot scene segmentation task, which is currently being implemented.

References

[1] Borkowski A., Dubrawski A., Racz J. Neural networks of ART type in navigation systems of mobile robots. *5th National Conference on Robotics*, Świeradów Zdrój, Poland, September 1996 (in Polish with English abstract).

[2] Box G.E.P., Draper N.R. *Empirical Model-Building and Response Surfaces*, Wiley 1987.

[3] 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.

[4] Dubrawski A., Schneider J. Memory-Based Stochastic Optimization for Validation and Tuning of Function Approximators. *6th Int. Workshop on Artificial Intelligence and Statistics AISTATS-97*, Fort Lauderdale, Florida, USA, January 1997.

[5] Kiefer J., Wolfowitz J. Stochastic estimation of the maximum of a regression function. *Annals of Mathematical Statistics*, 462–466, 23, 1952.

[6] Moore A.W., Lee M.S. Efficient Algorithms for Minimizing Cross Validation Error. *Proceedings of the 11th International Conference on Machine Learning*, 1994.

[7] Moore A.W., Schneider J. Memory based Stochastic Optimization. *Advances in Neural Information Processing Systems 8*, MIT Press, 1995.

[8] Moore A.W., Atkeson C.G., Schaal S. Memory-based Learning for Control. *Technical Report CMU-RI-TR-95-18*, Carnegie Mellon University, 1995.

[9] Racz J., Dubrawski A. Artificial neural network for mobile robot topological localization. *Robotics and Autonomous Systems*, 16(1), 73-80, 1995.