On Learning and Computational Complexity of FIR Radial Basis Function Networks, Part II: Complexity Measures

Kayvan Najarian
Computer Science Department, University of North Carolina at Charlotte
9201 University City Blvd., Charlotte, NC 28223, U.S.A.
E-mail: knajaria@uncc.edu

Abstract— **Recently, the complexity control of dynamic neural models has gained significant attention from signal processing community. The performance of such a process depends highly on the applied definition of "model complexity", i.e. complexity models that give simpler networks with better model accuracy and reliability are preferred. The learning theory creates a framework to assess the learning properties of models. These properties include the required size of the training samples as well as the statistical confidence over the model. In this paper, we apply the learning properties of two families of FIR Radial Basis Function Networks (RBFN's) to introduce new complexity measures that reflect the learning properties of such neural model. Then, based on these complexity term, we define cost functions, which provide a balance between the training and testing performances of the model, and give desirable levels of accuracy and confidence.**

Keywords— **Radial Basis Function Networks, Computational Complexity, Computational Learning Theory, PAC Learning, Finite Impulse Response (FIR) Models**

I. Introduction

The Probably Approximately Correct (PAC) learning theory, proposed by Valiant [1], deals with the accuracy and confidence of modeling tasks when data is i.i.d. Recently, new learning schemes are introduced that extend the results of the PAC learning to non-i.i.d. cases, and provide us with frameworks to assess learning properties of the dynamic modeling applications [2] with "Nonlinear Finite Impulse Response" (Nonlinear FIR) models. In the first part of this paper [3] using the extended PAC learning theory, the learning properties of FIR modeling using two families of Radial Basis Function Networks (RBFN's) were evaluated.

Despite the popularity of neural networks, overfitting is the main drawback of using such complex measures. The main challenge in minimum-complexity modeling of complex systems is to find relevant complexity terms to evaluate and limit the complexity of models. The complexity measures borrowed from the information theory and used in signal processing, such as the ones developed by Zhang and Muhlenbein (see for example [4]) do not directly address the testing-training balance and may not be the best choices for some neural modeling procedures. There is no evidence that the information theory based complexity measure (that mainly use Rissanen principle) are designed to create models with small training error and small Kolmogorov description length. However, there exists to clear mathematical equation (or inequality) that relates the closeness of training and testing error to Kolmogorov description length. It might be possible to create such a bridge by relating Kolmogrov description length to the Vapnik-Chervoninkis dimension [5], and as a result relate Kolmogorov description length to the overfitting problem; however this task seems to be a highly complicated one. Here, we use the results of the PAC learning theory to present new measures of complexity that target the overfitting issue directly and provides us with acceptable levels of the testing-training balance. Therefore, if the overfitting problem is the main concern in a modeling task, our complexity measures are more relevant than the Rissanen ones.

The paper is organized as follows: Par I of the paper [3] describes the basic definitions of the learning theory and gives the learning properties of FIR modeling using two families of Radial Basis Function Networks (RBFN's). In Section II, new complexity measures for the foresaid family of RBFN's are presented and the corresponding cost functions to be minimized during the training procedure are built. Section IV gives the results of the simulations performed using the proposed cost function and is followed by Section V which concludes the paper.

II. Model Complexity and Learning-Based Cost Function

Many practical analysis of the learning results on modeling applications reveal the inefficiency of the structural risk minimization algorithm in modeling of practical systems, where a limited size of data is available. This means that in a typical practical application, learning theory requires a huge training set to guarantee acceptable levels of accuracy and confidence, while in practice, only a limited number of training might be available. In other words, due to the conservative nature of the upper bounds available in the learning theory, the direct use of the structural risk minimization algorithm to many practical systems may not be possible. This observation holds not only for FIR modeling (using m-dependent data), but also static modeling with i.i.d. inputs. Here, we propose an approach that applies the results of the learning theory to perform a modeling task that provides us with some degree of confidence over the proximity of the testing and training errors. In order to do so, we introduce a complexity term and incorporate this term into the cost function to be minimized during the training.

As mentioned in Part I [3], given a fixed size of training data set, n, and a certain family of networks, PAC learning theory describes the deviation of the testing performance from the training performance by ϵ (i.e. the accuracy) and δ (i.e. the confidence). Based on PAC learning theory, in order to avoid overfitting in an ideal case, both these parameters must be as small as possible, at the same time. However, from the above inequalities, it can be seen that with a fixed number of training data points, reducing the value of the one of these two parameters would increase the bound on the other one. This means that with a fixed model and fixed number of training data, if higher accuracy is desired, one has to compromise the level of statistical confidence and vice versa. This issue parallels the bias-variance problem reported in the fields such as identification and modeling and asserts that often, both accuracy and confidence may not be minimized at the same time. To create a systematic modeling algorithm, the main objectives of a typical modeling task must be used with the learning properties to create a balance between these two parameters. In order to do this, one of the two parameters is fixed and the complexity measure is based on the other one. Notice that for a fixed level of disparity between the empirical error and the true error, δ gives the amount of uncertainty over the model. Therefore, it seems that for a typical modeling application, δ would be a good choice for a complexity measure to be minimized throughout the training process. Defining the complexity measure based on δ will result in complexity measures that are closely related to the level of smoothness (Lipschitz constant), the size of the training set as well as the dimension of the input. This is not surprising, because modeling with functions that possess higher Lipschitz constants and larger dimensionality is known to be more complex and to require more data points. Assuming that ϵ is constant, if δ (or a non-decreasing function of this parameter) is minimized during the training procedure, a model will be found for which the likelihood of having the difference between testing and training performances limited by ϵ is maximized. At this point, consider a set of Gausssian RBFN's. Similar procedures that give similar results for other neural structures are described later. Now, from Part I of this paper [3]:

$$\delta \geq 2^{\left[\frac{2\sqrt{2}A_{rbfn}d(\beta-\alpha)}{\epsilon\sqrt{e}}\right]^{d}}(m+1)\exp\left[-n\epsilon^{2}/8(m+1)\right] \ . \quad (1)$$

Assume that: $\alpha = 0$, $\beta = 1$. The choices of α and β are arbitrary and there is nothing special in the above choices. Also, note that the values of m, n, and δ are fixed during the training procedure. If the value of ϵ is given, the value of δ (or $\ln(\delta)$) is an indication of uncertainty of the model, i.e. minimizing $\ln(\delta)$ leads to models that are more reliable and perform more similarly on training and testing data sets. Therefore, one may define the complexity term C_{gauss} as $\ln(\delta)$. However, a closer look at the term A_{rbfn} is required. As defined in Part I of this paper, the term A_{rbfn} grades (nests) the parameter space. In other words, for a chosen value of A_{rbfn}, the space in which the parameters are allowed to vary is restricted. Then a higher value of A_{rbfn} defines a new parameter space which includes the previous space. In practice it is very difficult to create such a nested sequence of function sets based on a term such as A_{rbfn} and then search for the simplest function set with satisfactory performance. Also, performing minimization of the error for a fixed value of A_{rbfn} requires a solid method of constrained optimization to make sure that the resulting set of optimal parameters is indeed within the space specified by the prespecified value of A_{rbfn}. These limitations make the entire process of optimization within each of the function sets impractical. In order to obtain a sub-optimal solution, compromise is necessary. An optimization algorithm that is capable of minimizing non-smooth functions over the entire parameter space allows the use of a more practical complexity term which is formed by replacing A_{rbfn} with $\sum_i^l |a_i|\sqrt{b_i}$. During the consecutive iterations, the optimization algorithm now moves in favor of functions that fall within a smaller parameter space and as a result are less complex. This leads to the following more practical complexity term:

$$\begin{aligned} C_{guss} &= \left(\frac{2\sqrt{2}(\sum_i^l |a_i|\sqrt{b_i})d}{\epsilon\sqrt{e}} \right)^d \ln(2) \\ &+ \ln(m+1) - \frac{n\epsilon^2}{8(m+1)} \, . \end{aligned} \tag{2}$$

This complexity measure gives an indication of the testing-training balance and shows how unreliable the training of such a network is. It is important to notice that the above measure combines the parameter-space complexity with the complexity due to the number of neurons (structural complexity), without making any assumptions on the structure or parameters. This means that any algorithm that minimizes (2) addresses the parameter space complexity and the structural complexity at the same time. It can be observed that the complexity term introduced in (2) takes into consideration the size of available training data set n, the total dimension of input d, as well as the Lipschitz constant as an indication of smoothness of the function set. Now, suppose that during the training algorithm the number of neurons l is fixed. Then the objective of the optimization algorithm is to include both the above complexity term and the empirical error in a cost function, i.e. :

$$J_{guss} = \frac{1}{n}\left[\sum_{i=1}^{n} \boldsymbol{l}(f(x_i), y_i) \right] + \lambda C_{guss} \, . \tag{3}$$

In the above cost function, λ is a weighting factor that determines the relative importance of the empirical error and the complexity term in the overall cost function. Lower values of λ result in models that create small training errors but exhibit poor performance over the testing data. The higher values of λ create a desirable testing-training balance with both testing and training errors undesirably increased (due to resulting large empirical errors). The function $\boldsymbol{l}(.,.)$ is chosen to be a loss function that satisfies a uniform Lipschitz condition as defined in [6]. In the above cost functions, the values of $\sum_i^l |a_i|\sqrt{b_i}$ are kept as small as possible without being concerned about the over-all grading of the space through the fixed values of A_{rbfn}. Even with the relaxed definitions, since (3) is a non-differentiable function of the a_i's and perhaps of the b_i's, gradient-based minimization methods cannot be used for the optimization phase. Performing optimization over a_i's and b_i's thus calls for an algorithm that can minimize nonlinearly and non-smoothly parameterized models. In [7], elvolutionary programming is used to minimize similar inequalities, which can be extended to the optimization problem discussed above.

Next, following the same type of approach, complexity terms and cost functions are defined for the neural structures discussed above. Just as for Gaussian RBFN's, complexity terms are introduced that reflect the functional dependency of $\ln(\delta)$ and contain information about the learning properties of the modeling task. Starting with RMQ-RBFN's, use Inequality (7) in Part I and apply the logarithm on both sides to define the complexity term and the corresponding cost function as follows:

$$C_{rmq} = \left(\frac{4(\sum_i^l |a_i|\sqrt{b_i})d}{3\sqrt{3}\epsilon} \right)^d \ln(2)$$

$$+\ln(m+1) - \frac{n\epsilon^2}{8(m+1)} \,. \tag{4}$$

and:

$$J_{rmq} = \frac{1}{n}\left[\sum_{i=1}^{n} l(f(x_i), y_i)\right] \;+\; \lambda C_{rmq} \,. \tag{5}$$

In the above cost functions, λ is a weighting factor that determines the relative importance of the empirical error and the complexity term in the overall cost function. Lower values of λ result to models that create small training errors but might exhibit poor performance over the testing data. On the other hand, the higher values of λ create a desirable testing-training balance; however both testing and training errors might be too large.

It can be seen that the defined complexity terms depend not only on the number of hidden neurons, but also the size of the parameter space. This implies that the number of neurons can not be treated as the unique indication of model complexity. In some practical modeling algorithms, adding or deleting new neurons is the only way through which the complexity of a neural model is controlled. However, according to the above complexity term, it is possible that a model with fewer neurons have higher complexity due to its larger parameter space. This observation was first reported by Bartlett [8]. In [8], it is shown that the assumed size of the parameter space plays a more vital role in the complexity and reliability of a model than the number of neurons. This re-emphasizes the importance of considering the size of parameter space while adding or deleting neurons in designing variable-structure neural networks.

III. Simulation results

A number of simulations are performed in [9] to evaluate the performance of the introduced cost functions. Due to space limitation, a very brief description of the simulations is given here. In these simulations, a nonlinear function is modeled using RMQ-RBFN's with 10 hidden neurons. In the first simulation, the cost function contains only the empirical error ($\lambda = 0$). Minimizing this cost function results to the training and testing errors of 0.0854 and 0.0962, respectively. In the second simulation, the same training and testing data are used while the cost function contains both the empirical error and the complexity term ($\lambda = 1 \times 10^{-6}$). This simulation results in training and testing errors of 0.0887 and 0.0910, respectively. As can be seen, the difference between the training and testing errors in the second simulation is significantly smaller than that of the first simulation. This shows that the use of the introduced complexity terms in the overall cost function creates a balance between the training and testing errors and avoids overfitting.

IV. Conclusions

The cost function presented here applies the results of the learning theory and creates a balance between the empirical error and the complexity of RBFN's. Minimizing the novel complexity terms and cost functions introduced here result in models that avoid overfitting and create a suitable balance between the training and testing performances.

References

[1] L.G. Valiant, "A theory of learnable," *Comm. ACM*, pp. pp. 1134–1142, 1984.

[2] K. Najarian, Guy A. Dumont, and Michael S. Davies, "PAC learning in Nonlinear FIR Models," *submitted to: Journal of Adaptive Control and Signal Processing*, To appear.

[3] K. Najarian, "On Learning and Computational Complexity of FIR Radial Basis Function Networks, Part I: Learning of FIR RBFN's," *Submitted to ICASSP'2001*, May 2001.

[4] H. Muhlenbein, *Evolution and optimization '89. Selected papers on evolution theory, combinatorial optimization, and related topics. Papers from the International Workshop on Evolution Theory and Combinatorial Optimization held in Eisenach*, Akademie-Verlag, Berlin, 1990.

[5] V.N. Vapnik and A.Y. Chervonenkis, "On the uniform convergence of relative frequencies of events to their probabilities," *Theory of Probability and its Applications*, vol. 16, no. 2, pp. 264–280, 1971.

[6] K. Najarian, G.A. Dumont, M.S. Davies, N.E. Heckman, "Learning of FIR Models Under Uniform Distribution," *Proc. The American Control Conference, San Dieo, U.S.A. (ACC1999)*, pp. 864–869, June 1999.

[7] K. Najarian, G.A. Dumont and M.S. Davies, "A learning-theory-based training algorithm for variable-structure dynamic neural modeling," *Proc. Inter. Joint Conf. Neural Networks (IJCNN99)*, 1999.

[8] P. Bartlett, "The sample complexity of pattern classification with neural networks: the size of the weights is more important than the size of the network," *Amer. Statistical Assoc. Math. Soc. Transactions*, vol. 17, pp. 277–364, 1996.

[9] K. Najarian, *Appliation of learning theory in neural modeling of dynamic systems*, Ph.D. thesis, Dpartment of Electrical and Computer Engineering, University of British Columbia, 2000.