CNNA-94 Third IEEE International Workshop on Cellular Neural Networks and their Applications
Rome, Italy, December 18-21,1994

Computational Learning Theory Applied to Discrete-Time Cellular Neural Networks

Wolfgang Utschick and Josef A. Nossek
Institute for Network Theory and Circuit Design
Technical University of Munich,
Arcisstr. 21, 80333 Munich, Germany
Phone: +49-89-2105-8520,
Fax: +49-89-2105-8504,
Email: wout@nws.e-technik.tu-muenchen.de

Abstract *In this paper the theory of probably approximately correct (PAC) learning is applied to Discrete-Time Cellular Neural Networks (DTCNNS). The Vapnik-Chervonenkis dimension of DTCNN is to be determined. Considering two different operation modes of the network, an upper bound of the sample size for a reliable generalization of DTCNN architecture will be given.*

1 Introduction

An important aspect of neural networks is generalization, i.e., the ability of the networks to correctly deal with input data which were not included in their training data. One of the most important issues in that context of learning from examples is the *sample complexity* that gives an upper bound of samples sizes required for reliable generalization of the neural network.
In the field of Computational learning theory [1] there are many types of models for "learning". A very usefull mathematical model within a probabilistic framework of learning and generalization is the "probably approximately correct" (PAC) learning theory introduced by Valiant [10]. Within the PAC theory the *expressive power* of networks, i.e, the ability to realize arbitrary mappings on the input space plays an important role. Using methods based on PAC learning Baum and Haussler [3] analysed the generalization abilities of feedforward networks of linear threshold elements and found upper bounds for samples sizes. They show that for $0 \leq \epsilon \leq \frac{1}{2}$, if a sample of size $m \geq m_0 = (\frac{32W}{\epsilon})\ln(\frac{32N}{\epsilon})$ is loaded into a feedforward network of linear threshold elements with N nodes and W weights, so that a fraction of at most $\frac{\epsilon}{2}$ of the examples are not correctly classified, then with confidence of at least $1-8e^{-1.5W}$ the network will correctly classify all but a fraction ϵ of future examples.
Standard PAC learning theory applies only to Boolean-valued functions or to classification tasks corresponding to (multilayer) feedforward networks with binary single output. A number of extensions and variations on the basic PAC model have been made [2]. There are extensions for analyzing feedforward linear threshold networks having more than one output node [9] or artificial neural networks with a real-valued output.

0-7803-2070-0/94/$4.00 © 1994 IEEE

In the next section we will give a brief description of the PAC learning theory applied to neural networks. Due to the geometrical properties and constraints of DTCNNs we obtain a bound for the expressive power of this type of Cellular Neural Networks in section 3. In section 4 we derive the sample complexity of DTCNNs, i.e. an upper bound of the sample size that guarantees a certain generalization of the network.

2 PAC Learning

PAC learning represents a probabilistic framework for learning from examples in the field of neural networks. In this framework there is a given set of inputs and a space of functions called *hypotheses space* H_{net} that is within the scope of the given network. There is assumed to be an unknown *target concept* $t : X \rightarrow Y$ from the *input space* to $\{0,1\}$. The goal of learning is to produce an appropriate adjustment of the weights of the neural network that realizes a good approximation called *hypothesis* h of the unknown underlying *target concept* t. For the derived hypothesis h to have a predictive power there must be a relation between the training and testing set of examples, i.e., in PAC learning the *test sample* has to be drawn according to the same probability distribution μ as the *training sample*. Formally spoken, the input space is a probability space (X, μ) and the hypotheses space H_{net} is a set of measurable functions from X to $\{0,1\}$. The target concept t is assumed to be element of H_{net}. A *learning algorithm* is one that takes the examples and produces the hypothesis. For the subsequent outline of the theory we make some useful definitions.

Definition 1: *Let the training sample* s *for the target concept* t *with sample size* m *defined by*

$$\mathrm{s} = ((\mathbf{x}_1, t_1), (\mathbf{x}_2, t_2), \cdots, (\mathbf{x}_m, t_m)) \in S(m,t) \subset (X \times \{0,1\})^m,$$

$$t_i = t(\mathbf{x}_i).$$

Definition 2: *The observed error of the trained network (hypothesis) on the training sample with respect to the target concept is given by*

$$er_s(h,t) = \frac{1}{m}|\{i \in \{1,\cdots,m\} \mid h(\mathbf{x}_i) \neq t(\mathbf{x}_i)\}|.$$

Definition 3: *Consequently the error of the trained network on the total input space with respect to the target concept equals*

$$er_\mu(h,t) = Prob_\mu\{\mathbf{x} \in X \mid h(\mathbf{x}) \neq t(\mathbf{x})\}.$$

A characteristic property of an neural network architecture is the maximum possible number of different classifications $\Pi_H(m)$ that can be realized by the chosen network for a sample with a given size.

Definition 4: *The corresponding growth function is defined by*

$$\Pi_H(m) = \max_{\mathbf{x} \in X} |\{(h(\mathbf{x}_1), h(\mathbf{x}_2), \cdots, h(\mathbf{x}_m)) \in \{0,1\}^m \mid h \in H_{net}\}|.$$

With the definition of the growth function we are able to give the subsequent theorem according to the sample complexity of reliabe generalization.

Theorem 1: *Suppose that H_{net} is the hypotheses space of all mappings that can be performed by the neural network on the input space (X, μ), and that t, μ and ϵ are arbitrary but fixed. Then*

$$Prob_{\mu}^{m}\{s \in S(m,t) \mid \textit{for all } h \in H_{net},\ er_s(h,t) = 0 \Rightarrow er_{\mu}(h,t) \leq \epsilon\} \geq 1 - 2\Pi_H(2m)2^{-\epsilon m/2}$$

$$\textit{for all positive integers } m \geq \frac{8}{\epsilon},$$

i.e., if no error can be observed on the training sample of size m (sample complexity) after training the network the probability of an error of at most ϵ on future examples tends to certainty under certain assumptions of the growth function.

Proof: in [4, 1].

The growth function $\Pi_H(m)$ depends on the set of hypotheses that can be realized by the given neural network, i.e., it depends on the topology of the network and the activation function of its nodes and the weights of its connections between nodes. There is a surjection from the space of possible weight assignments to the space of hypotheses. It is clear, that for any sample size m the growth function is bounded by

$$\Pi_H(m) \leq 2^m,$$

i.e., the number of possible binary functions on m patterns. An appropriate measure of the expressive power of a network architecture is the value of the sample size m_{VC} that turns out to be the largest one at which the network still has the power to induce all 2^m binary functions. Hence the *Vapnik-Chervonenkis dimension* m_{VC} [2] is defined as

Definition 5: *Vapnik-Chervonenkis dimension is given by*

$$m_{VC} = \max\ m \ \text{ subject to } \ \Pi_H(m) = 2^m \ \wedge \ \Pi_H(m+1) < 2^{m+1}.$$

If there is no finite m_{VC} the VC dimension is called infinite. If a neural network has an infinite VC dimension it is not learnable in the sense of PAC learning theory [2]. For finite VC dimension the following result holds [8]:

$$\textit{for all } m \geq m_{VC} \qquad \Pi_H(m) \leq \left(\frac{em}{m_{VC}}\right)^{m_{VC}}.$$

The definition of the VC dimension helps to introduce a more sophisticated notation of theorem 1 [1].

The sample complexity is the least value $m_0(\delta,\epsilon)$ *such that for all target concepts* t *and probability distributions* μ *and* $0 \leq \delta, \epsilon,$

$$Prob_{\mu}^{m}\{\mathrm{s} \in S(m,t) \mid \textit{for all } h \in H_{net}\ ,\ er_s(h,t) = 0 \Rightarrow er_{\mu}(h,t) \leq \epsilon\} \geq 1-\delta$$

$$\textit{whenever} \quad m \geq m_0(\delta,\epsilon).$$

$$\textit{It holds} \quad m_0(\delta,\epsilon) \leq \frac{4}{\epsilon}\left\lceil m_{VC} \lg \frac{12}{\epsilon} + \lg \frac{2}{\delta} \right\rceil .$$

3 Expressive Power of DTCNNs

The Discrete-Time Cellular Neural Network (DTCNN) [5] is a discrete-time version of the CNN. Its characteristic properties are the architectural features of CNN, translationally invariant weights, local interconnections, binary output property and a *signum* activation function. The DTCNN is a nonlinear discrete-time first-order dynamical system and can be viewed as a special case of a standard Hopfield model with a parallel update strategy. Let M be the number of cells on the cell grid of the DTCNN. Then $\mathrm{u}(t) \in [-1,+1]^M$ and $\mathrm{y}(t) \in \{-1,+1\}^M$ denote the vector of the input signals and the output vector of the cells. The vector of the cell states $\mathrm{x}(t) \in R^M$ is defined by the state equation,

$$\begin{aligned} \mathrm{x}(t+1) &= \mathbf{A}\mathrm{y}(t+1) + \mathbf{B}\mathrm{u}(t+1) + \mathrm{i} \\ \mathrm{y}(t+1) &= \mathrm{sgn}(\mathrm{x}(t+1)). \end{aligned}$$

A more compact notation for the operation of the network at cell level is the *cell level notation* that relates to the view of a single cell as a standard perceptron architecture. Introducing the vectors e_c and p the cell output of a cell can be written by the *local transition function* f,

$$\mathrm{y}_c(t+1) = \mathrm{sgn}(\mathbf{p}^T \mathrm{e}_c(t)),$$

where $\mathrm{e}_c(t)$ comprises the total number of input lines for a single cell c including the input signals $\mathrm{u}_c(t)$ of the cell itself and the output signals $\mathrm{y}(t)$ of nearby cells, due to the recurrent structure of the network. By analogy with $\mathrm{e}_c(t)$ the vector p comprises the corresponding weights of all input signals of the cell. Each of the vectors is assumed to have N elements. The details are omitted here. For more information see [6]. The global mapping properties of the network refer to the mappings performable by a DTCNN from the input signals $\mathrm{u}(t)$ and the initial states y_0 of the cells to the outputs $\mathrm{y}(T)$ at a fixed time-step T. For analyzing the expressive power of the DTCNN the number of performable global mappings is of particular interest. An independent criteria for the expressive power is the VC dimension of the network architecture.

Theorem 2: *The VC dimension of a DTCNN is given by*

$$\text{VCdim} \leq N + 1.$$

Proof: Due to the translational invariance of the templates and the perceptron-like architecture of the DTCNN cells, the number of performable mappings by a DTCNN can be bounded by the number of mappings performable by a single *N-input linear threshold element*. Consequently, the growth function (VC dimension) of the lineaer threshold element is an upper bound for the growth function (VC dimension) of the DTCNN. The VC dimension of a N-input linear threshold element is defined by $N + 1$ [2].

Obviously, only the *number* of performable mappings is compared with the corresponding quantity of a lineaer threshold element. Due to the recurrence of the DTCNN architecture, a DTCNN can realize more sophisticated mappings then a simple perceptron architecture.

4 Sample Complexity of DTCNNs

The mapping properties of the DTCNN critically depend on the mode of operation of the network. There are two interesting cases which are suitable for the analysis of its sample complexity. In both cases the DTCNN constitutes a mapping from one multiple space to another. The multiple nature of the mapping does not affect the results of the PAC learning. See also [9]. In the first case we consider the global mapping of the network from a constant input vector $\mathbf{u}(0)$ and the initial state $\mathbf{y}(0)$ to an output vector $\mathbf{y}(T)$ after T time-steps.

Remark 1: *The sample complexity of learning the global mapping* $(\mathbf{u}(0), \mathbf{y}(0)) \rightarrow \mathbf{y}(T)$ *of a DTCNN does not depend on the number of time-steps of the trajectory and can be written as*

$$m_0(\delta, \epsilon) \leq \frac{4}{\epsilon}\left[(N+1)\lg\frac{12}{\epsilon} + \lg\frac{2}{\delta}\right].$$

The second case deals with learning a trajectory of the output vector $\mathbf{y}(t)$ through the time axis for T time steps. Due to the discrete-time nature and the parallel update strategy of the DTCNN the learning of a trajectory can be split into a problem of T *global transitions* of the M cells. Therefore, taking into account the translationally invariant properties of the DTCNN the learning environment already can be identified at the level of local transitions of the cells.

Remark 2: *The sample complexity of learning a trajectory* $(\mathbf{u}(0), \mathbf{u}(1), \cdots, \mathbf{u}(T-1), \mathbf{y}(0)) \rightarrow (\mathbf{y}(1), \mathbf{y}(2), \cdots, \mathbf{y}(T))$ *of the output vector* $\mathbf{y}(t)$ *for a sequence of the input vector* $\mathbf{u}(t)$ *and the initial state* $\mathbf{y}(0)$ *of the DTCNN refers to the reliable amount of trajectories and is given by*

$$m_0(\delta, \epsilon) \leq \frac{4}{\epsilon}\left[\frac{1}{T \cdot M}\left((N+1)\lg\frac{12}{\epsilon} + \lg\frac{2}{\delta}\right)\right].$$

5 Conclusion

Applying the *probably approximately correct learning theory* to DTCNN we have derived an upper bound of the number of examples for reliable generalization when learning from examples. Due to the perceptron-like architecture of the DTCNN the *Vapnik-Chervonenkis dimension* of this type of neural network has been given. The VC dimension of the DTCNN is defined by $N+1$, for N being the number of inputs for each cell of the network. Considering a neighborhood size of r, N can be written as $N = 2(2r+1)^2$. Obviously, the local interconnection structure of the DTCNN results in a lower sample complexity of the learning problem. Remark, that the results critically depend on the assumption that the test sample has to be drawn according to the same probability distribution as the training sample. This has to be taken into account when analyzing applications like in [7].

References

[1] M. Anthony and N. Biggs. *Computational Learning Theory*, volume 30 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, 1992.

[2] M. Anthony and N. Biggs. Computational Learning Theory for Artificial Neural Networks. In J. G. Taylor, editor, *Mathematical Approaches to Neural Networks*, pages 25–62. Elsevier Science Publishers B.V., 1993.

[3] E. B. Baum and D. Haussler. What Size Net Gives Valid Generalization? *Neural Comp.*, 1:151–160, 1989.

[4] A. Blumer, A. Ehrenfeucht, and et al. Learnability and the Vapnik-Chervonenkis Dimension. *Journal of the ACM*, 36:929–965, 1989.

[5] H. Harrer and J.A. Nossek. Discrete-Time Cellular Neural Network. *International Journal of Circuit Theory and Applications*, 20:453–467, 1992.

[6] H. Magnussen. *The Discrete-Time Cellular Neural Network: Further Properties and Global Learning Algorithms*. PhD thesis, Technische Universität München, Lehrstuhl für Netzwerktheorie und Schaltungstechnik, 1994. to be published.

[7] P. Nachbar, J. Strobl, and J.A. Nossek. Kantenextraktion aus Grauwertbildern mit DTCNN. *Kleinheubacher Berichte*, 37:239–244, 1993.

[8] N. Sauer. On the density of families of sets. *Journal of Combinatorial Theory (A)*, 13:145–147, 1972.

[9] J. Shawe-Taylor and M. Anthony. Sample sizes for multiple output threshold networks. *Network*, 2:107–117, 1991.

[10] L.G. Valiant. A theory of the learnable. *Communications of the ACM*, 27:1134–1142, 1984.