A FEEDFORWARD NETWORK OF LEARNING AUTOMATA FOR PATTERN CLASSIFICATION

M.A.L. Thathachar and V.V. Phansalkar
Department of Electrical Engineering
Indian Institute of Science
Bangalore 560 012

ABSTRACT

A model made of units of teams of learning automata is developed for the three layer pattern classifier. The algorithm is approximated by an ODE, using weak convergence methods. The pattern recognition problem is posed as a constrained maximization problem. It is shown that the zeros of the ODE correspond to points satisfying first order necessary conditions of the maximization problem. Partial results are obtained in showing that the ODE ,and hence the algorithm, converges to the local maxima of the maximization problem.

1. Introduction

Pattern recognition(PR) has been widely studied as a method for classifying objects into categories [3]. One of the earliest methods, which has been a focus of renewed interest, is the three layer network implementing piecewise linear discriminant functions [2,3]. The first layer of the network is composed of units which implement hyperplanes. The pattern vector is input to each of these units and they output a 0 or 1 depending on whether the pattern is on the positive or negative side of the hyperplane being implemented by the unit. The second layer is composed of units which implement the AND function. The inputs to the second layer units are the outputs of the first layer units. The third layer is composed of a single unit which ORs the second layer outputs. This network is for a two class problem. It can easily be generalized to a multiclass problem. There are various ways of implementing the units in the network [3]. Here, units composed of teams of Learning Automata (LA) are considered. Details about LA can be found in [4]. The first layer is composed of M units. The pattern vector is of dimension m. Each first layer unit is a team of m LA. U_i is the ith first layer unit and A_{ij} its jth LA. The pattern vector is $\underline{x}$, where

$$\underline{x} = (x_1, x2, \ldots, x_m)^t \tag{1.1}$$

and $x_m = 1$ is added for thresholding. Let $\underline{w}_i$ be a m-dimensional real vector. Define y_i as

$$y_i = us(\underline{w}_i^t \underline{x}) \tag{1.2}$$

where us(.) is the unit step function and

$$\underline{w}_i = (w_{i1}, \ldots, w_{im})^t \tag{1.3}$$

At instant k A_{ij} chooses a particular value, out of a set S_{ij} of prefixed values, for w_{ij}. Let S_{ij} have L elements. Thus,

$$S_{ij} = \{v_{ij1}, \ldots, v_{ijL}\} \tag{1.4}$$

and A_{ij} picks up value v_{ijn} for w_{ij} with probability p_{ijn}. Thus,

$$\Pr\{w_{ij}(k) = v_{ijn}\} = p_{ijn}(k) \tag{1.5}$$

and

$$\underline{p}_{ij} = (p_{ij1}, \ldots, p_{ijL})^t \tag{1.6}$$

where $\underline{p}_{ij}$ is the probability vector of A_{ij}. y_i is the output of U_i. The second layer is composed of N units. V_i is the ith second layer unit. Assume that V_i has inputs from alf first layer units. This simplifies the notation , and the case in which second layer units have inputs only from some first layer units can be handled identically. With the above assumption, each second layer unit has M LA, each LA with two actions, 0 and 1. B_{ij} is the jth automaton of V_i. Let z_{ij} be the output of B_{ij}. As B_{ij} has two actions, its probability vector has one component, q_{ij}. Then,

$$\Pr\{z_{ij}(k) = 1\} = 1 - \Pr\{z_{ij}(k) = 0\} = q_{ij}(k) \tag{1.7}$$

a_i is the output of V_i. a_i is the AND of all y_j such that $z_{ij} = 1$.

The third layer output is denoted by z. This is the OR of all the z_is. There is no learning involved in this layer.

For the PR problem, denote the two classes by 0 and 1. There has to be a figure of merit to determine the efficacy of the network. One of the most widely used is to minimize the probability of misclassification [3], $P(e)$, where

$$P(e) = \Pr\{\text{class } 0\}\Pr\{z=1|\text{class } 0\} + \Pr\{\text{class } 1\}\Pr\{z=0|\text{class } 1\} \tag{1.8}$$

At instant k, $\underline{x}(k)$ is the pattern vector and $z(k)$ the classifier output. A signal. $r(k)$ is generated as follows

$$r(k) = 1 \quad \text{if } \underline{x}(k) \text{ classified correctly; 0 else} \tag{1.9}$$

The signal $r(k)$ is broadcast to all the LA. Depending on $r(k)$ and the action taken, each LA updates its probability vector according to the learning algorithm. The updating is decentralized, each LA needing to know only its own action and the value of $r(k)$. Maximizing the expected value of r is the same as minimizing $P(e)$. It should be noted that no assumption is made as to the structure of the class distributions or values of the a priori probabilities. It is only assumed that the pattern vectors arrive according to the a priori probability values and class distributions. Section 2 develops the learning algorithm and analysis. The simulation results are presented in section 3 and the conclusions in Section 4.

2. ALGORITHM AND ANALYSIS

In this section, the algorithm is described and analyzed.

The algorithm used is the L_{RI} [4].

First Layer: For all (i,j,n) admissible

If $w_{ij}(k)=v_{ijn}$, $p_{ijn}(k+1) = (1 - br(k))\, p_{ijn}(k) + br(k)$ (2.1a)

Else, $p_{ijn}(k+1) = (1 - br(k))\, p_{ijn}(k)$ (2.1b)

Second Layer: For all (i,j) admissible

If $z_{ij}(k)=1$, $q_{ij}(k+1) = (1 - br(k))\, q_{ij}(k) + br(k)$ (2.1c)

Else, $q_{ij}(k+1) = (1 - br(k))\, q_{ij}(k)$ (2.1d)

where $0 < b < 1$ is the learning parameter. Different learning parameters can be used for different LA.

Let $(\underline{p},\underline{q})$ be the vector composed all the action probability vectors, $\underline{p}$ for the first layer and $\underline{q}$ the second layer. An ideal system should converge to a $(\underline{p}, \underline{q})$ such that for each pattern vector, the output should be the correct class w.p.1. This may not be possible due to limitations imposed by network structure. A more feasible goal is to maximize $f(\underline{p},\underline{q})$ where

$$f(\underline{p},\underline{q}) = E[r(k) \mid \underline{p}(k) = \underline{p},\ \underline{q}(k) = \underline{q}] \tag{2.2}$$

such that p_{ijn} and q_{ij} remain probabilities. That is,

$$\text{maximize } f(\underline{p},\underline{q}) \tag{2.3a}$$

$$\text{subject to } g_{ijn}(\underline{p},\underline{q}) = p_{ijn} \geq 0 \tag{2.3b}$$

$$h_{ij}\ (\underline{p},\underline{q}) = \sum_{n} p_{ijn} - 1 = 0 \tag{2.3c}$$

$$s_{ij}\ (\underline{p},\underline{q}) = q_{ij}\ (1 - q_{ij}) \geq 0 \tag{2.3d}$$

The asymptotic behaviour of 2.1, for small b, is approximated by the asymptotic behaviour of a related ODE, obtained from 2.1 using weak convergence techniques [1]. To obtain the ODE, the following lemma is required. The <u>drifts</u> Δp_{ijn} and Δq_{ij} are defined by

$$\Delta p_{ijn} = E[p_{ijn}(k+1) - p_{ijn}(k) \mid \underline{p}(k) = \underline{p}, \underline{q}(k) = \underline{q}] \tag{2.4a}$$
$$\Delta q_{ij} = E[q_{ij}(k+1) - q_{ij}(k) \mid \underline{p}(k) = \underline{p}, \underline{q}(k) = \underline{q}] \tag{2.4b}$$

Lemma 2.1: The drifts Δp_{ijn} and Δq_{ij} are

$$\Delta p_{ijn} = bc_{ijn}(\underline{p}, \underline{q}) = bp_{ijn}(\delta f/\delta p_{ijn} - f) \tag{2.5a}$$

$$\Delta q_{ij} = bd_{ij}\ (\underline{p}, \underline{q}) = bq_{ij}\ (1 - q_{ij})\ (\delta f/\delta q_{ij}) \tag{2.5b}$$

Comment: The proof is by using the fact that the network is a feedforward network and by repeated conditioning.

For each $b > 0$, $\{\underline{p}(k),\underline{q}(k)\}$ is a Markov process. Denote this by $\{\underline{p}^b(k),\underline{q}^b(k)\}$. Define continuous time interpolations $\{\underline{P}^b(.),\underline{Q}^b(.)\}$ of $\{\underline{p}^b(.),\underline{q}^b(.)\}$ as

$$\underline{P}^b(t) = \underline{p}^b(k) \text{ and } Q^b(t) = \underline{q}^b(k) \text{ if } t \in [kb, (k+1)b) \qquad (2.6)$$

Theorem 2.1: The sequence of interpolated processes, $\{\underline{P}^b(.), \underline{Q}^b(.)\}$, converges weakly as $b \to 0$, to $(\underline{\alpha}(.), \underline{\beta}(.))$ where

$$\dot{\alpha}_{ijn} = c_{ijn}(\underline{\alpha}, \underline{\beta}); \quad \underline{\alpha}(0) = \underline{p}(0) \qquad (2.7a)$$

$$\dot{\beta}_{ij} = d_{ij}(\underline{\alpha}, \underline{\beta}); \quad \underline{\beta}(0) = \underline{q}(0) \qquad (2.7b)$$

Comment: Here $\underline{\alpha}$ corresponds to $\underline{p}$ and $\underline{\beta}$ to g. The proof follows from theorem 5.5.2 in [5].

It can be shown that the asymptotic behaviour of 2.1, for small b, is similar to the asymptotic behaviour of 2.7. Thus, only a study cf the relationship between the maximization problem 2.3 and ODE 2.7 need be done. The ideal relationship which can be expected is as follows

locally stable zero <==> local maximum (2.8a)
locally asymptotically stable <==> strict local maximum (2.8b)
no limit cycles for the ODE (2.8c)

The above conditions would imply that the ODE, and hence algorithm 2.1, converges to a local maximum of 2.3. The following results are directed towards proving 2.8.

The First Order Necessary Kuhn Tucker (FONKT) conditions are necessary conditions to be satisfied by any $(\underline{p}, \underline{q})$ which is a local maximum of 2.3 [5]. For 2.3, the FONKT conditions are

$$\begin{aligned} &\delta f/\delta p_{ijn} + \lambda_{ijn} + \mu_{ij} = 0; \quad \lambda_{ijn}\, g_{ijn} = 0 \\ &\delta f/\delta q_{ij} + \xi_{ij}\,(1 - 2q_{ij}) = 0; \quad \xi_{ij}\, s_{ij} = 0 \qquad (2.9) \\ &\lambda_{ijn} \ge 0; \quad g_{ijn} \ge 0, \quad \xi_{ij} \ge 0; \quad s_{ij} \ge 0; \quad h_{ij} = 0 \end{aligned}$$

for all admissible i,j,n. It can be shown that all zeros of 2.7 satisfy 2.9, else they are unstable. First, a result concerning the behaviour of f(.) will be stated.

Lemma 2.2: (df/dt) is always non-negative and (df/dt) $(\underline{p}, \underline{q})$ is zero iff $(\underline{p}, \underline{q})$ is a zero of 2.7.

Comment: The proof follows by using the chain rule and rearranging terms to get a sum of positive terms. It can be shown that all these positive terms are zero only at a zero of the ODE 2.7. It is seen. from the above result that there can be no limit cycles for 2.7, as f(.) would have to strictly increase along the limit cycle, which is impossible. This proves 2.8c.

Lemma 2.3: If $(\underline{p},\underline{q})$ satisfies FONKT conditions 2.9, then $(\underline{p},\underline{q})$ is a zero of ODE 2.7. Conversely, if $(\underline{p},\underline{q})$ is a zero of the ODE and does not satisfy the FONKT conditions, $(\underline{p},\underline{q})$ is an unstable zero.

Comment: It can be shown that $(\underline{p}, \underline{q})$ satisfying 2.8 would imply that each positive term mentioned in the proof of Lemma 2.2 is zero. Thus, df/dt is zero and by Lemma 2.2, $(\underline{p}, \underline{q})$ is a zero of the ODE. For the converse, the only way 2.9 is not satisfied is by atleast one of the λ_{ijn}, ϵ_{ij} being strictly negative. It can be shown, using linear approximation, that the particular component which has a strictly negative λ_{ijn} or ϵ_{ij} is unstable. This result shows that the only zeros of interest are those which satisfy 2.9. A local minimum of f(.) cannot satisfy 2.9, as the FONKT conditions for local minimum are different [5]. The next result is a partial proof of 2.8a and 2.8b.

Lemma **2.4**: Let M be compact, connected and f = K on M. Let M be a strict local maximum of the maximization problem 2.3. Then M is a locally stable attractor of the ODE 2.7. Conversely, let M be a locally asymptotically stable attractor of the ODE 2.7. Then M is a strict local maximum of the optimization problem 2.3.

Comment: The proof is direct using Lyapunov techniques. The converse proof uses lemma 2.2 and properties of asymptotically stable attractors. Lemma 2.4 goes partially towards proving 2.8a and 2.8b. It is difficult to prove 2.8a and 2.8b due to the fact that stability implies only the existence of a Lyapunov function. Also, if the ODE is linearized around a zero, the linear part can have zero eigenvalues from which no conclusions can be drawn.

3. SIMULATION RESULTS

In this section, an example is chosen and simulated. The pattern vectors, $\underline{x} = (x_1, x_2, x_3)^t$, are assumed to arrive randomly from the set $X = [0,1] \times [0,1] \times \{1\}$. The classes are 0 and 1. X is divided into two regions A and B. B consists of $\underline{x}$ such that

$$[x_1 > 3/41 \vee \{[x_1 > 1/41 \wedge [x_2 > 1/2]\} \tag{3.1}$$

is true. The rest of X is region A. In region A,

$$\text{Prob}\{\underline{x} \in \text{class } 0\} = 1 - \text{Prob}\{\underline{x} \in \text{class } 1\} = 0.8 \tag{3.2}$$

and in region B the values are interchanged. Thus, to minimize P(e), $\underline{x}$ in A should be classified as class 0 and in B as class 1.

The network is composed of three first layer units, U_1, U_2 and U_3; and two second layer units V_1 *and* V_2. All second layer units are connected **to** all first layer units. Each first layer unit has three LA, and each LA has five actions, being the discretized weight values. w_{ij} is the action of A_{ij}. w_{ij} takes values from $S_{ij} = \{v_{ij1}, v_{ij2}, v_{ij3}, v_{ij4}, v_{ij5}\}$. **Table** 1 gives the first layer **unit details** and Table 2 the second layer unit details. Automata A_{i3} of the first layer units is for thresholding. **33** simulation runs were done. Since the results in Section 2 are local results, an initial bias towards the optimal actions is required. For the first layer the initial probability of the optimal action was set at 0.6 and for the second layer at 0.7. The learning parameter for the first layer was 0.005 and 0.001

for the second layer. Convergence to the optimal set of actions was observed in 30 cases. Average number of steps to reach a probability of 0.99 for all the actions in the optimal set was 62,600. This excludes the three wrong convergences.

4. CONCLUSIONS

It is shown that LA can be used as units in feedforward networks. The ease of analysis of such networks is seen. Simulations confirm the analytical results. The pattern recognition problem considered here is the supervised pattern recognition problem. Samples with known classes are assumed to arrive according to unknown a priori probabilities and class distribution. No information is assumed about these. The pattern recognition problem has been posed as an optimization problem which is related to an ODE derived from the algorithm.

5. REFERENCES

1. H.J.Kushner: Approximation and weak convergence methods for random processes, with applications to stochastic systems theory, MIT Press, Cambridge, 1984.
2. R.P.Lippmann: An introduction to computing with neural nets, IEEE ASSP Magazine, April 1987, pp. 4-12.
3. W.S.Meisel: Computer oriented approaches to pattern recognition, Academic Press, New York, 1972.
4. K.S. Narendra and M.A.L. Thathachar: Learning Automata - An Introduction, Prentice Hall, Englewood Cliffs, 1989.
5. W.Zangwill: Nonlinear Programming - An Unified Approach, Prentice Hall, Englewood Cliffs, 1969.

Unit	Automaton	Actions					Index of optimal action
		1	2	3	4	5	
	A_{11}	-1	-1/2	0	1/2	1	5
U_1	A_{12}	-1	-1/2	0	1/2	1	3
	A_{13}	-1	-1/2	-1/4	0	1/2	3
	A21	-1	-1/2	0	1/2	1	3
U_2	A22	-1	-1/2	0	1/2	1	5
	A_{23}	-1	-1/2	0	1/2	1	2
	A_{31}	-1	-1/2	0	1/2	1	5
U_3	A_{32}	-1	-1/2	0	1/2	1	3
	A33	-1	-3/4	0	0	1/2	2

Table 1

Unit	(Automaton,Optimal action)
V_1	$(B_{11},1),(B_{12},1),(B_{13},0)$
V_2	$(B_{21},\text{any}),(B_{22},0),(B_{23},1)$

Table 2