The Generalization Capabilities of ARTMAP

Gregory L. Heileman
Department of Electrical and Computer Engineering
University of New Mexico, Albuquerque, NM 87131
heileman@eece.unm.edu

Michael Georgiopoulos
Department of Electrical and Computer Engineering
University of Central Florida, Orlando, FL 32816
mng@ece.engr.ucf.edu

Michael J. Healy
The Boeing Company
P.O. Box 3707 MS 7L-66, Seattle, WA 98124
mjhealy@boeing.com

Stephen J. Verzi
Department of Computer Science
University of New Mexico, Albuquerque, NM 87131
verzi@cs.unm.edu

Abstract

Bounds on the number of training examples needed to guarantee a certain level of generalization performance in the ARTMAP architecture are derived. Conditions are derived under which ARTMAP can achieve a specific level of performance assuming any unknown, but fixed, probability distribution on the training data.

1. Introduction

A common framework for studying machine learning assumes a *training phase* in which a learning system (or learner) is allowed to study a set of labeled training examples (i.e., a training sample), and is then asked to produce a hypothesis that in some sense "explains" the underlying process that created these examples. A subsequent *performance phase* occurs when the hypothesis produced by the learner is used in an application to explain unlabeled examples. The degree to which the hypothesis succeeds during this later phase is often referred to as its *generalization capability*. Since the performance phase is where a learning machine is actually used to solve problems, its ability to produce hypotheses with good generalization capability is by far its most important feature. In many cases, this ability is estimated empirically by testing an output hypothesis against previously unseen labeled examples. In this paper, however, we are concerned with deriving analytical results that give a guarantee on the generalization capability of ARTMAP.

A number of previous papers have dealt with the performance and capabilities of ARTMAP networks during the training phase (e.g., [3]). This paper is concerned with the derivation bounds on the number of training examples required in order to guarantee a certain level of generalization performance in ARTMAP during the performance phase.

Specifically, let us assume ARTMAP is given a training sample of the form $S = \{(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)\}$, where x_i is an example and y_i is the label of this example. During the training phase, ARTMAP modifies its weights according to this training data. We can think of the final configuration of the network, after the training phase, as a hypothesis. The goal during training is to produce a hypothesis h such that $h(x_i) \approx y_i$ for all i. In many cases, the y_i's may take on only a fixed number of values. Each of these possible values can be thought of as a *class*; in which case, h is performing classification. When each y_i may take on only one of two possible values (i.e., classes), it is nat-

0-7803-4122-8/97 $10.00

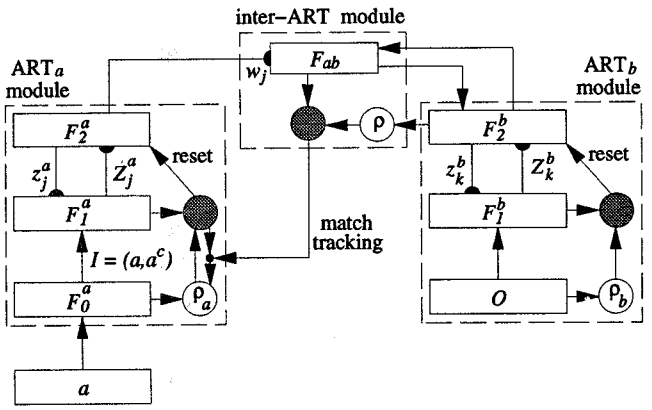

Figure 1. A block diagram of the ARTMAP architecture.

ural to talk about examples as being either positive or negative examples of some unknown target concept c. In this case, h itself can be thought of as a concept, and the learner is said to be performing a special type of inductive learning called *concept learning*.

Any hypothesis that correctly classifies *all* examples in a training sample S is called a *consistent hypothesis* for S. An important property of ARTMAP, not shared by most neural network models, is that very simple conditions exist under which it is guaranteed to produce consistent hypotheses. This property will be used to derive bounds on the generalization performance of ARTMAP in the following sections. Specifically, in Section 2 we present the ARTMAP model that we will analyze. The notion of generalization capability is then presented in a more formal setting when we discuss the Probably Approximately Correct (PAC) learning model in Section 3. The generalization results we obtain for ARTMAP are then presented in Section 4. In Section 5 we consider how the results of this paper can be applied.

2. The ARTMAP Architecture

The ARTMAP architecture, which is described in detail by Carpenter et al. [2], is a neural network that can be used to learn arbitrary mappings from a Boolean input space of any dimensionality, to a Boolean output space of any dimensionality.

The ARTMAP neural network consists of two ART1 modules designated as ART$_a$ and ART$_b$, as well as an inter-ART module as shown in Figure 1. Inputs (examples) are presented at the ART$_a$ module, while outputs (labels) are presented at the ART$_b$ module. The inter-ART module includes a MAP field, whose purpose is to determine whether the mapping between the presented inputs and outputs is the desired one. If $a = (a_1, \ldots, a_{M_a})$ denotes a binary vector, the input to the ART$_a$ module is the binary vector

$$I = (a, a^c) = (a_1, \ldots, a_{M_a}, a_1^c, \ldots, a_{M_a}^c)$$

where

$$a_i^c = 1 - a_i \quad 1 \leq i \leq M_a$$

This type of transformation, called *complementary coding*, is necessary for the successful operation of ARTMAP (for more details see [2], page 584). A field of nodes designated as F_0^a receives the input vector a and produces the input I for the ART$_a$ architecture. Hence, F_0^a acts as a preprocessor to the ART$_a$ module. The binary input vector I is subsequently applied at the F_1 field of ART$_a$, designated as F_1^a. No such transformation (i.e., complementary coding) is necessary for the output O which is directly applied at the F_1 field of ART$_b$, denoted as F_1^b. Field F_1^a has $2M_a$ nodes, field F_1^b has M_b nodes, the F_2 field of ART$_a$ (F_2^a) has N_a nodes, the F_2 field of ART$_b$ (F_2^b) has N_b nodes, and finally the MAP field F_{ab} has N_b nodes. Fields F_2^a and F_2^b are where compressed representations of the input patterns (the I's) and the output patterns (the O's) are established, respectively. Certain other minor assumptions about the model are necessary in order to derive our results. These assumptions are all satisfied by the fast learning ARTMAP architecture discussed in Georgiopoulos et al. [3].

In this paper we consider concept learning. In particular we wish to use ARTMAP to learn a concept corresponding to an unknown target concept c that performs a mapping from n-dimensional Boolean space to 1-dimensional Boolean space, i.e., $c : \{0,1\}^n \longrightarrow \{0,1\}$, where the 1-bit output is 0 for a negative example of the concept, and 1 for a positive example. In order for ARTMAP to function properly, the vector $(0,1)$ will be supplied to the ART$_b$ for the labels of a negative examples, and the the vector $(1,0)$ will be supplied for the labels of positive examples. Our final modeling assumption is that the number of F_2^a nodes needed during training is $O(m)$. This assumption is certainly reasonable in light of the fact that if more than $O(m)$ such nodes are needed, the network is not acting to compress the training sample during training. We will use $\mathcal{AM}$ to refer to an ARTMAP architecture that satisfies all of the modeling assumptions discussed above, and $\mathcal{AM}_n$ when we wish to indicate that $M_a = n$ in this architecture.

It is assumed that a sample of size m is available for training $\mathcal{AM}_n$. Each element of this sample consists of a randomly drawn vector in $\{0,1\}^n$, along with a label, $(0,1)$ or $(1,0)$, corresponding to the output that c produces on this vector. This paper addresses the ability of ARTMAP to generalize. Specifically, we will consider the question of what sample size m is necessary for training $\mathcal{AM}_n$ so that

we have a confidence δ that it will correctly classify (according to c) a fraction $1-\epsilon$ of future randomly drawn unlabeled vectors in $\{0,1\}^n$.

3. The PAC Model

One of the most popular and widely studied theoretical models of concept learning is the Probably Approximately Correct (PAC) model introduced by Valiant [4]. A reasonable assumption regarding this model is that the more examples processed during the training phase, the better the generalization capability of the resulting hypothesis will be.

In the PAC model we assume examples are drawn according to distribution $\mathcal{D}$ from the instance space X. It is convenient to think of the the elements of X as being parameterized on some parameter n. In which case, we can write $X = \cup_{n\geq 1} X_n$. If $C = \cup_{n\geq 1} \mathcal{C}_n$ is a class of concepts defined over X, and $c \in \mathcal{C}_n$ is an unknown target concept (i.e., the one we are trying to learn), then the error of the hypothesis h output by our learning algorithm is defined as

$$error_{\mathcal{D}}(h) = \Pr_{x\in\mathcal{D}}\{h(x) \neq c(x)\},$$

where the subscript $x \in \mathcal{D}$ indicates that the probability is taken with respect to a random draw of $x \in X_n$ according to $\mathcal{D}$. Given this definition, we are now ready to define PAC learning.

The concept class $\mathcal{C}$ is said to be PAC learnable if there exists an algorithm L such that for every $c \in \mathcal{C}_n$ over any distribution $\mathcal{D}$ on X, and for all $0 < \epsilon, \delta < 1/2$, if L is given access to labeled examples and inputs ϵ and δ, then with probability at least $1-\delta$, L outputs a hypothesis $h \in \mathcal{H}$ satisfying $error_{\mathcal{D}}(h) \leq \epsilon$. Furthermore, if L runs in time polynomial in n, $1/\epsilon$, $1/\delta$, and the size of c, then we say $\mathcal{C}$ is *efficiently* PAC learnable by hypothesis class $\mathcal{H}$.

If the concept class is defined over n-dimensional Boolean space, as is the case for the ARTMAP architecture $\mathcal{AM}_n$, then explicit conditions under which PAC learning occurs are given in the following theorem. In this theorem, $|\mathcal{C}| \subseteq 2^{|X|}$ denotes the total number concepts in any finite concept class $\mathcal{C}$.

Theorem 1 *(Blumer et al. [1])*
Let $\mathcal{C}$ be any finite concept class. Then for any $0 < \epsilon, \delta < 1/2$, given a training sample of size

$$m \geq \frac{1}{\epsilon}\ln\left(\frac{|\mathcal{C}|}{\delta}\right),$$

drawn independently and at random according to $\mathcal{D}$ and classified according to concept class $c \in \mathcal{C}$, with probability at least $1-\delta$, any hypothesis h consistent with the training sample satisfies $error_{\mathcal{D}}(h) \leq \epsilon$.

Thus, if an ARTMAP network is able to exactly learn a large enough training sample, then it will PAC learn the unknown target concept c.

4. ARTMAP Results

In order to derive generalization results for the ARTMAP architecture $\mathcal{AM}$, we will make use of the following important theorem:

Theorem 2 *(Georgiopoulos, et al. [3])*
Given a training sample for the ARTMAP architecture $\mathcal{AM}_n$, $\mathcal{AM}_n$ will consistently learn the training sample after the entire training sample has been presented at most $(n-1)$ times.

We are now ready to state our main result, which quantifies the generalization capability of the ARTMAP architecture $\mathcal{AM}$.

Theorem 3 *Any Boolean concept class $\mathcal{C}$ is efficiently PAC learnable using the ARTMAP architecture $\mathcal{AM}$ if $|\mathcal{C}_n| = O(b^{n^k})$, where b and k are positive constants.*

PROOF: Assume c is any concept in $\mathcal{C}_n$. Choose $0 < \epsilon, \delta < 1$, and draw a sample of size

$$m \geq \frac{1}{\epsilon}\ln\left(\frac{|\mathcal{C}|}{\delta}\right).$$

Next, construct the ARTMAP network $\mathcal{AM}_n$, and repeatedly present the training sample to the network $(n-1)$ times. According to Theorem 2, after this training, $\mathcal{AM}_n$ will be consistent with all m training examples. Since m was chosen according to Theorem 1, the hypothesis h produced by $\mathcal{AM}_n$ satisfies.

$$\Pr_{x\in\mathcal{D}}\{error_{\mathcal{D}}(h) \leq \epsilon\} \geq 1-\delta.$$

Since we have assumed the number of F_2^a nodes is $O(m)$, and $\ln b^{n^k} = O(n^k)$, the training time can be upper bounded by $m^2(n-1) = O(n^{k+1})$. Because k is a constant, the running time is therefore polynomial in n, $1/\epsilon$, $1/\delta$, and the size of the concept. $\diamondsuit$.

In the following section we use Theorem 3 to explore the generalization capabilities of ARTMAP for a number of interesting problems.

5. Examples

A *monomial* formula defined on n Boolean variables is any purely conjunctive collection of these variables or their complements. The number of different monomials that can be defined on n variables is 3^n. Therefore, if we

are asked to learn some unknown monomial concept over 20 variables to 95% accuracy ($\epsilon = 0.05$) with 99% confidence ($\delta = 0.01$), Theorem 3 tells us that this can be done using $\mathcal{AM}_{20}$ and a sample of size $m = 532$.

A *disjunctive normal form* (DNF) formula defined on n Boolean variables is any collection of monomial clauses defined on these variables that are joined together by disjunction. A k-DNF Boolean formula is a DNF Boolean formula in which each monomial clause may contain at most k entries. The space of k-DNF formulae is much richer than the space of monomial formulae. Specifically, it can be shown that the number of different k-DNF formulae that can be defined on n variables is at most $2^{(2n)^k}$. Thus if we are asked to learn some unknown 10-DNF concept over 20 variables to 95% accuracy with 99% confidence, Theorem 3 tells us that this can be done using $\mathcal{AM}_{20}$ and a sample of size $m = 554,517$. If, however, we assume the concept can be represented by some k-DNF formula using a small number of clauses, say p, then the number of different formulae that can be defined on n variables is at most $p(2n)^k$. In this case, for $p = 10$, $\mathcal{AM}_{20}$ would require a sample of size $m = 73,777$.

It is interesting to compare these results to those obtained by training ARTMAP on some real-world data sets. Carpenter et al. [2] performed an extensive series of experiments using ARTMAP on a benchmark machine learning database known as the mushroom database. Each training example in this database is a 126-element binary feature vector, along with a classification as to whether the mushroom under consideration is edible or poisonous. A total of 23 species are represented in the database. With $\overline{\rho}_a = 0.7$ and off-line training (which is the type of training considered in this paper), they report an average accuracy of 97.7% (over 10 runs) using a training sample of size 1000. Furthermore, Table 5 on pg. 576 of their work demonstrates that the hypothesis output by ARTMAP is actually a k-DNF formula. From their table, it can be seen that the maximum size of any clause is 15. From our previous example, however, we see that Theorem 3 would direct us to use a much larger sample size. It remains an open question as to what extent the bounds in Theorem 3 can be tightened.

Acknowledgments: This research was supported by a grant from The Boeing Company under contract W-300445.

References

[1] A. Blumer, A. Ehrenfeucht, D. Haussler, and M. K. Warmuth. Learnability and the Vapnik-Chervonenkis dimension. *Journal of the ACM*, 36(4):929–965, 1989.

[2] G. A. Carpenter, S. Grossberg, and J. H. Reynolds. ARTMAP: Supervised real-time learning and classification of nonstationary data by a self-organizing neural network. *Neural Networks*, 4(5):565–588, 1991.

[3] M. Georgiopoulos, J. Huang, and G. L. Heileman. Properties of learning in ARTMAP. *Neural Networks*, 7(3):495–506, 1994.

[4] L. G. Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, 1984.