On Learning Boolean Functions

B. K. Natarajan

CMU-RI-TR-86-17

The Robotics Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

December 1986

Table of Contents

Abstract

This paper deals with the learnability of boolean functions. An intuitively appealing notion of dimensionality of boolean functions is developed and used to identify the most general class of boolean function families that are learnable from polynomially many positive examples with one-sided error. It is then argued that although bounded DNF expressions lie outside this class, they must have efficient learning algorithms as they are well suited for expressing many human concepts. A framework that enables efficient learning of bounded DNF functions is then identified.

1- Introduction

Much attention has been paid to machine learning in the area of Artificial Intelligence (Michalski, Carbonnell and Mitchell, 1983). Most of the literature in this category report investigations involving heuristic and ad hoc methods for solving specific problems. Until recently, formal approaches were limited to the area of inductive inference, as surveyed in (Angluin and Smith, 1983). The work reported in (Valiant, 1984) and (Blumer, Ehrenfeucht, Haussler and Warmuth, 1986) has helped refocus interest in a formal framework for the problem.

(Valiant, 1984) presents computational models for learning and derives algorithms for learning specific classes of boolean functions like bounded CNF, monotone DNF etc. In answer to an important question left open in (Valiant, 1984), we identify a general family of functions that is no harder to learn than bounded CNF but is not expressible as k-CNF for any fixed k. Going further, we give a theorem that identifies the most general class of function families that can be learnt with one-sided error from polynomially many positive examples. In doing so we develop the notion of dimensionality for boolean functions, in a manna: that is independent of the structure of their boolean expressions. This is the main result of this paper. Our development of the notion of dimensionality was independant of that first introduced in (Vapnik and Chervonenkis, 1971) and more recently discussed in (Blumer et al, 1986). Although our notion is less general, in that it is specific to boolean functions, it is far more intuitive than that of (Vapnik and Chervonenkis, 1971). The supporting lemmas and theorems in this paper do not invoke the heavy machinery that (Blumer et al, 1986) borrow from (Vapnik and Chervonenkis, 1971), as they (the lemmas and theorems) are no stronger than necessary to the problem at hand. For the sake of completeness, we show that our notion of dimension is equivalent to that of (Vapnik and Chervonenkis, 1971)*

We then argue that many human concepts are bounded DNF, and hence there must be efficient algorithms that learn them. Using our dimensionality theorem we show that bounded DNF functions cannot be learnt with one-sided error from polynomially many positive examples. Placing some restrictions on the probability distribution of the examples allows us to obtain a polynomial time algorithm that leams bounded DNF expressions with two-sided "error. We can show that learning algorithms chat work for arbitrary

distributions do not exist. This leads us to believe that while human concepts are often bounded DNF, they are not learnt as such, but as disjunctions of simpler sub-concepts.

2. Preliminaries

We consider n boolean variables $v_1,...v_n$ that can take on values from {0,1}. An *assignment* is an assignment from {0,1} to each of the variables. A *learning algorithm* is an algorithm that attempts to infer a function f on the variables, from positive examples for f. i.e, assignments that satisfy f. The learning algorithm has at its disposal a subroutine EXAMPLE, that at each call produces a positive example for the function to be learnt. For a particular assignment a, the probability that the learned function will be queried on a is $P(a)$, as given by the probability distribution function P. Also, the probability that a particular satisfying assignment a of f will be produced by a call of EXAMPLE is $P(a)/(\sum P(a_p))$, where the summation is over the set of satisfying assignments for f. In addition the learning algorithm knows a priori that the function to be learned belongs to some subset of all boolean functions on the same variables.

We define a *family* of functions F to be any set of boolean functions such that if f and g are two functions in F with the same number of variables, then they are also functions of the same variables. The n^{th} *subfamily* F_n of a family F is the set of functions in F of n variables. Hence

$$F = \cup_{n \geq 1} F_n.$$

3. Learning with One-Sided Error

Following (Valiant, 1984), we say that a family of functions F is learnable with one-sided error if and only if there exists an algorithm that

(a) makes polynomially many calls of EXAMPLE both in an adjustable error parameter h and in the number of variables n. The algorithm need not run in polynomial time.

(b) for all functions f in F, and all probability distributions P over the assignments, the algorithm deduces with probability $(1-1/h)$ a function g in F such that for any assignment a,

$g(a)=1$ implies $f(a) = 1$

if $S = \{a \mid f(a) = 1 \text{ and } g(a) = 0\}$, then

$$\sum_{a \in S} \wedge \quad * \quad y\ h$$

Further, if the algorithm runs in polynomial time, we say that the family is polynomial time learaable or p-time leamable. The notions of learnability and p-time learnability capture the idea that the teacher's time is a lot more valuable than the student's time. So a concept is leamable if it requires a small amount of student-teacher interaction, even though the student may have to do a lot of homework.

(Valiant, 1984) shows that bounded CNF expressions are p-time leamable with one-sided error and questions whether the class of p-time leamable functions can be significantly extended beyond the bounded CNF functions. We will exhibit a rather general family of functions that is p-time leamable with one-sided error but is not included in the bounded CNF families. For any positive integer m, let ZT^{l} be the family of boolean functions whose $/1^{th}$ sub-family consists of all functions of *n* variables with at most *ti** satisfying assignments.

Theorem 1: For any *m, BT* is p-time leamable with one-sided error*

Proof: (sketch) Construct an algorithm that calls EXAMPLE *iKtf" + hgji)* times and presents as output the disjunction of the distinct assignments produced by EXAMPLE. With probability (1—I/A) the algorithm will output a function in If^{l} that differs from the function to be learned with probability no greater than *TJk* This can be proved with arguments similar to that of (Valiant, 1984). •

Theorem!: For any fixed *m* and *k*, $\bar{B}r \leq L\ k\sim CNR$

Proof: Consider the function^ of *n* variables v^v^ given by

$$f_n = (v_1 \vee v_2 \cdots \vee v_N) \wedge v_{N+1} \cdots \wedge v_n$$

where *N= mlogn.* Now / has $2^{OT\&8''} = «^{m}$ assignments and so *f* is in *EP*. Suppose that for some *k*, *f* is expressible as k-CNF for any *n.* Pick *n* such that *mlogn* > *k* and examine the clauses of the k-CNF fonnula g_{f^*} that is supposedly equivalent to /, $g_{a\ ft}$ must contain a clause of the form

$$(u_x \vee u_r \vee u_t \vee \neg u_{i+l} \vee \cdots \neg jij)$$

where $0 < / \prec$ it and

$\{u_1,...,u_i\} \subseteq \{v_1,...,v_N\}$

$\{u_{i+1},...,u_k\} \subseteq \{v_{N+1},...v_n\}$

If not, g_n has a satisfying assignment that sets all of $v_1,...,v_N$ to 0 and hence $g_n \neq f_n$. But then, every satisfying assignment of g_n has atleast one of $u_1,...,u_i$ set to 1 and again $g_n \neq f_n$. Hence the theorem. □

It is easy to see that the above proof holds for the family of functions with at most $X(n)$ satisfying assignments where $X(n)$ is an increasing function. Furthermore, any such family is p-time learnable as well, extending the class of p-time learnable functions well beyond the k-CNF expressions.

Next we identify the set of all function families that are learnable with one-sided error. We need the following definitions for our discussion. In what follows, we shall use the name of a function to refer to the function or the set of assignments that satisfy it, unless the context demands clarification.

A boolean function is said to be *consistent* with a set of assignments if it is satisfied by every assignment in the set.

A sub-family F_n is *well-ordered* if, for any set of assignments S that is consistent with some function in F_n, there exists a function f in F_n such that

(a) f is consistent with S

(b) for any $f_1 \in F_n$ consistent with S, $f \subseteq f_1$

We say that f is the least function in F_n consistent with S. A family F is well-ordered if its n^{th} sub-family F_n is well-ordered for all n.

Define the operator M_n to be a mapping from sets of assignments to functions in F_n as follows. For any subset S of assignments consistent with some f in F_n, $M_n(S)$ is the least function in F_n consistent with S. If S is not consistent with any function F_n, $M_n(S)$ is undefined.

Proposition 1: F_n is well-ordered if and only if for any two functions f and g in F_n, $f \cap g \neq \emptyset$ implies that there exists a function $h \in F_n$ such that $h = f \cap g$.

Proposition 2: For any two sets *A* and 5,

$$M_n(A \cup 5) = M_n(M_n(A) \cup M£B))$$

The *dimension* of a sub-family F^ denoted by dimC/y, is the least integer *d* such that for every function/ in fT, there exists a set 5-of *dor* fewer assignments such that/is the least function in F_n consistent with S_f.

A family *F* is of polynomial dimension if there exists a polynomial *D(n)* such that for all $n_{\%}$ F_n is of dimension at most *D(n).*

Proposition 3: Let F_n be of dimension *d* Then there exists a set *S* of *d* assignments such that

(a) *S* is consistent with some function in F_n.

(b) for any $S_{1\$}S_2 \subseteq S$,

$$S_x \neq £ S_2 \text{ implies } MJiSJ \neq y\& \ ^{M}£^{s})$$

Proof: Since F_n is of dimension *d,* there is some function/in *F^* such that for any set of assignments *S*

/= $M_n(S)$ implies |S| ≥ *d*.

Pick a set *S* of rf^^ignments sach that/= $M_n(S)$ and (a) is satisfied

Then, suppose that S_x and S_2 are two subsets of *S* such that S_l 7^ $S_{\%}$ and yet $MJiS^ = M^(S_2)$. Without lo^ of generality let $\backslash S_X\backslash \leq \backslash S_2 l$ Now,

$$S = (S - S_2) \cup S_1 \cup S_2$$

Using Proposition 2 on the above equation we get

$$M_n\{S) = MJ.MJLS\text{-}SJ \cup M^) \cup M_n(S_2))$$

$$= M_n(M_n(S\text{-}SJ \cup MJiSJ) \text{ since } M_n(S_1) = M_n(S_2).$$

$$= M_n((S\text{-}SJ \cup 5^) \text{ by Proposition 2 again.}$$

$$\text{Hence/= } M_n(S) = MJ£S - S_2) \cup S_1)$$

where |(5 - S p u ^ l < < *L*

A contr^IkrtioiL Hence the *remit* D

Theorem 3: A family F is learnable with one-sided error if and only if F is of polynomial dimension.

Proof:(sketch) (only if) We begin by showing that if a family is learnable with one sided-error, it is well-ordered. Let F be a family that is learnable, but not well-ordered. Let A be a learning algorithm for F_n. Then, for any set of assignments that is consistent with some function in F_n, A produces the least such function. Otherwise A would not learn with one-sided error. Hence F_n is well-ordered and so is F.

Let A be an algorithm that learns F_n, using $(n/h)^k$ calls of EXAMPLE, where k is some integer and h is the error parameter. Suppose that F_n is of super-polynomial dimension $D(n)$. Then, pick n and h such that $D(n) > (n/h)^k(1-h)$. Let $d = D(n)$ and $m = (n/h)^k$. Recall that A must learn any function in F_n for any probability distribution on it. Pick a set S of size d as in Proposition 3, and place the uniform probability distribution on it. Now, A will see some m elements of S and output the least function g that is consistent with these. By Proposition 3, this function will not be consistent with any of the other d-m elements in S. Consequently, g differs from the function to be learned, $M_n(S)$, with greater than $1/h$ probability. Hence A does not learn F_n.

(if) Let F be of polynomial dimension $D(n)$. We build an algorithm A to learn F as follows. Let $f \in F_n$ be the function to be learned with error at most $1/h$ as defined earlier. For any probability distribution P on f, consider the set of functions

$$C_f = \{g | g \in F_n, P(f-g) \geq 1/h\}$$

These are the functions that differ from f with more than the allowable error. The probability that m calls of EXAMPLE will produce assignments all consistent with some particular function in C_f is bounded by $(1-1/h)^m$. Now $|C_f| \leq |F_n| \leq 2^{nD(n)}$. Hence the probability that all m examples will be consistent with any one function in C_f is bounded by $2^{nD(n)}(1-1/h)^m$. Therefore, if

$$2^{nD(n)}(1-1/h)^m < 1/h$$

is satisfied, A will learn F. Simplifying, we get

$$m > h(nD(n) + \log n)$$

to be sufficient. Since $D(n)$ is polynomial, m is polynomial in n and h, making F learnable with one-sided

error.

This completes the proof. □

Corollary: A boolean family F is learnable with one-sided error if and only if there exists a polynomial $D(n)$ such that

$$|F_n| \leq 2^{D(n)} \text{ for all } n.$$

We will now work an example to demonstrate the scope of Theorem 3.

Example 1: For each n, given are $k(n)$ predetermined boolean functions $f_1 f_2 \ldots f_{k(n)}$, where $k(n)$ is some fixed polynomial in n. Consider the function family F whose n^{th} subfamily F_n is defined as follows.

$$F_n = \{g | g = \bigwedge_{f_i \in S} f_i, S \subseteq \{f_1, f_2, \ldots f_{k(n)}\}\}$$

In words, F_n consists of conjuncts of any subset of the given $k(n)$ functions.

Claim 1: F is well-ordered.

Proof: Straightforward. □

Claim 2: F is of polynomial dimension.

Proof: For any function f in F_n, construct a set of assignments S_f as follows.

begin

$S_f = \emptyset$

for each g of the given $k(n)$ functions **do**

if g is not included in f, **pick an**

assignment that satisfies f **but not** g.

If such exists, add it to S_f.

od

end

S_f contains at most $k(n)$ elements and it is easy to verify that $f = M_n(S_f)$. Hence F_n is of dimension $k(n)$ and therefore F is of polynomial dimension. □

Claim 3: F is learnable with one-sided error.

Proof: Immediate from Theorem 3 and the foregoing claims. □

Indeed, it is not hard to see that bounded CNF is a special case of this family. □

Theorem 3 is useful in determining whether or not a family of functions is learnable. However, it cannot be used to determine p-time learnability. This is precluded by generality of the theorem in that it makes no assumptions on the structure of the function family. To extend the the scope of the theorem to p-time learnability, we need the following definition.

A sub-family F_n is *orderable* in time t if and only if for any set of assignments S that is consistent with some function in F_n, the least function in F_n consistent with S is computable in time t. A family F is p-time orderable if and only if there exists a polynomial $T(n)$ such that for all n, F_n is orderable in $T(n)$ time.

Theorem 4: A family F is p-time learnable if and only if it is of polynomial dimension and is p-time orderable.

Proof: A straightforward extension of the proof of Theorem 3. □

We will now establish the relationship between our notion of dimension and that of (Vapnik and Chervonenkis, 1971). Following (Blumer et al, 1986), we define the Vapnik-Chervonenkis dimension, denoted by $d_{vc}(F_n)$, of a subfamily F_n as follows.

Given a set S of assignments of n variables, let $\Pi(S)$ denote the set of all subsets of S obtained by intersecting S with the functions in F_n, i.e.

$$\Pi(S) = \{S \cap f | f \in F_n\}$$

If $\Pi(S) = 2^S$, we say S is shattered by F_n. $d_{vc}(F_n)$ is the smallest integer d such that no set of cardinality $d+1$

is shattered by F_n.

Theorem 5: For any subfamily F_n,

$$dim(F_n) \leq d_{vc}(F_n) \leq n \cdot dim(F_n)$$

Proof: Let $\dim(F_n) = d$. Since $|F_n| \leq 2^{nd}$, no set of cardinality $nd+1$ can be shattered by F_n. Hence,

$$d_{vc}(F_n) \leq n \cdot dim(F_n)$$

By Proposition 3, there exists a set of cardinality d that is shattered by F_n. Hence,

$$dim(F_n) \leq d_{vc}(F_n)$$

and therefore

$$dim(F_n) \leq d_{vc}(F_n) \leq n \cdot dim(F_n)$$

□

4. Learning with Two-Sided Error

Consider the family of boolean functions generated by the bounded-DNF expressions. i.e., for any k, the k-DNF functions are those represented by DNF formulae with at most k variables per clause. It is easy to verify that for any fixed k, the k-DNF functions are not well-ordered. Take a single assignment. There are many k-DNF functions that are consistent with this assignment, but there is no least one among them. Hence k-DNF functions are not learnable with one-sided error. This is unfortunate as many human concepts tend to be bounded DNF rather than CNF, perhaps because the former naturally favours positive concepts like

$$(this \wedge that \wedge that) \vee (that \wedge that) \vee ...$$

while k-CNF favours negative concepts like

$$(\neg(this \wedge that \wedge that)) \wedge (\neg(that \wedge that)) \wedge ...$$

In words, k-DNF expresses what is included by the concept, while k-CNF expresses what is excluded by it.

Suppose we send out a child to buy some fruit. Given his limited development, let us assume that the child only knows of apples, mangoes, pomegranates and grapes. So his concept of fruit is likely to be

$$fruit = apples \vee mangoes \vee pomegranates \vee grapes$$

$$= (red \wedge juicy \wedge fleshy) \vee (yellow \wedge juicy \wedge fleshy) \vee$$

$$(red \wedge \neg juicy \wedge cellular) \vee (purple \wedge juicy \wedge fleshy)$$

where each clause in the second expression describes the corresponding fruit in the first, in terms of the observable variables. It appears that the concept of fruit is better expressed in DNF rather than CNF. Also, k-DNF concepts tend to be general, in that any k-DNF formula has atleast 2^{n-k} satisfying assignments.

The point of all this is that bounded-DNF functions are important and since they are learned easily by humans, there must be efficient algorithms for them. Our interest here is best expressed thus. Suppose the child in the above example delegated the task to the family robot. How does he get the robot to learn quicky what he means by "fruit"? In particular, we wish to find the framework within which the child can quickly communicate to the robot what he means by a fruit, as opposed to analyzing a particular protocol with respect to such concepts. We will now define such a framework.

A family of functions F is learnable with two sided error if and only if there exists a learning algotithm that

(a) makes polynomially many calls of EXAMPLE both in the adjustable error parameter h and in the number of variables n.

(b) For all functions f in F_n, and the uniform distribution over the assignments, the algorithm will deduce with probability $(1-1/h)$ a function g in F such that for any assignment a,

$$P(g(a) \neq f(a)) \leq 1/h$$

We emphasize here that the uniform probability distribution applies to the examples produced by EXAMPLE as well as to the queries asked of the learned function. If the distribution is allowed to be arbitrary, we can show easily that any learning algorithm can be forced to deviate from the function to be learned with probabilit approaching unity.

We say a k-DNF clause is *prime* if and only if it contains k distinct variables. (A variable and its negation are not distinct). We now present an algorithm that learns k-DNF functions of n variables.

Algorithm 1

(1) Perform $(32/z^3*/*^3)$ calls of EXAMPLE and for each prime k-DNF clause c_1 of the n variables, compute the hit frequency

$$r_1 = \frac{\text{number of examples satisfying } c_f}{\text{total number of examples}}$$

(2) Pick the clause C_m such that r_m is a maximum. Construct g, the disjunction of all clauses c_f such that

$$r_i \geq r_m - \frac{1}{2h(2n)^k}$$

(3) Output the constructed function g.

Define the *hit probability* of a clause to be the probability that a call of EXAMPLE will result in a satisfying assignment of the clause. Algorithm 1 does nothing more than estimate the hit probability of each prime k-DNF clause. (There are no more than $(^2£) \leq (2n)^k$ of these.) Since the probability distribution is uniform, all of the prime clauses that are included in the function to be learned will enjoy the same hit probability and that will be a maximum. Hence, the algorithm attempts to identify the clauses with maximum hit probability and will do so with high probability. There might be some clauses that are not included in the function to be learned, but have a hit probability approaching the maximum. These will be included in the output function g if their error contribution is lower than that allowed by the parameter k Some algebraic manipulation is required to compute the number of calls to EXAMPLE as specified in the algorithm. In particular, an approximation of the binomial distribution (Feller, 1957) is useful in estimating the number of calls to EXAMPLE necessary to ensure that for any clause, with proabability at least (1-1/A), the deviation of its hit frequency from its hit probability is at most $S = \frac{1}{-}$. This ensures that with probability (1—I/A),

(a) the clauses in the function to be learned will enjoy a hit frequency in the interval $(r_m - 28)$, where r_m is the maximum hit frequency observed.

(b) a clause with tût probability less than (^—45) will have a hit frequency outside the interval $(r_m, r_m - 2\delta)$.

Of the above conditions, (a) implies that with high probability, our algorithm will correctly identify the clauses present in the function to be learned, while (b) implies that the error contribution of a spurious clause that is included by fee algorithm in the output function is at most 45. Since there can be at most $(2n)^k$ spurious functions, it follows that with high probability the output function is incorrect with probability at most

$$(2/0*45 = \frac{4(2n)^k}{4h(2n)^k} = 1/h.$$

Which is as required.

From &e forgoing, we see that bounded DNF functions are not easy to learn except in some restricted situations. Perhaps humans do not learn k-DNF functions at one go, but do so clause by clausele, bottom up instead of top-down. In the robot-child example, this translates to the child teaching the robot tiie sub-concepts of apple, mango, grape and pomegranate, and then unifying them into the concept of fruit. Of course, if negative examples were allowed, bounded DNF functions are no harder to learn than bounded CNF functions. But it is unusual for human concepts to be learned from negative examples.

5. Conclusion

In this paper, we dealt with the teamabiBty of boolean functions. We identified the most general class of Sanction families that are leamable with one-sided error from polynomially many positive examples. In doing so, we developed an intuitively appealing notion of dimensionlity for boolean functions. Arguing that many human concepts are bounded DNF, we identified a framework within which bounded DNF expressions are **tamable,**

6. References

Ansbb, O. & Smith* **GIL (1983).** *Crnnmtring Surveys,* Vol 16, No *X* Sept, pp237-269.

Bhmer, A., Ehitafcucbt A*, Hinder, O, & Wmniith, M. (1986). *ACM Symposium on Theory cf* **Computing, pp273-2JEL**

Feller* W. (1957). in *foimductkm* 10 *Probability Theory ami Us Applications* * John Wiley and Sons.

Michalski, R.S., Carbonnell, J.G., & Mitchell, T.M. (1983). *Machine Learning: An Artificial Intelligence Approach*, Tioga Publishing Co., Palo Alto, CA.

Valiant, L.G. (1984). *ACM Symposium on Theory of Computing*, pp436-445.

Vapnik, V.N. & Chervonenkis, A.Ya. (1971). *Theory of Probability and its Applications*, Vol 16, p264-280.