, , 1–50 ()
© Kluwer Academic Publishers, Boston. Manufactured in The Netherlands.

On the Complexity of Function Learning

PETER AUER pauer@igi.tu-graz.ac.at
PHILIP M. LONG plong@igi.tu-graz.ac.at
WOLFGANG MAASS maass@igi.tu-graz.ac.at
GERHARD J. WOEGINGER gwoegi@igi.tu-graz.ac.at
Institute for Theoretical Computer Science
Technische Universität Graz
Klosterwiesgasse 32/2
A-8010 Graz, Austria

Editor: Sally A. Goldman

Received May 1, 1991

Abstract. The majority of results in computational learning theory are concerned with concept learning, i.e. with the special case of function learning for classes of functions with range $\{0,1\}$. Much less is known about the theory of learning functions with a larger range such as $\mathbb{N}$ or $\mathbb{R}$. In particular relatively few results exist about the general structure of common models for function learning, and there are only very few nontrivial function classes for which positive learning results have been exhibited in any of these models.

We introduce in this paper the notion of a binary branching adversary tree for function learning, which allows us to give a somewhat surprising equivalent characterization of the optimal learning cost for learning a class of real-valued functions (in terms of a max-min definition which does not involve any "learning" model).

Another general structural result of this paper relates the cost for learning a union of function classes to the learning costs for the individual function classes.

Furthermore, we exhibit an efficient learning algorithm for learning convex piecewise linear functions from $\mathbb{R}^d$ into $\mathbb{R}$. Previously, the class of linear functions from $\mathbb{R}^d$ into $\mathbb{R}$ was the only class of functions with multi-dimensional domain that was known to be learnable within the rigorous framework of a formal model for on-line learning.

Finally we give a sufficient condition for an arbitrary class $\mathcal{F}$ of functions from $\mathbb{R}$ into $\mathbb{R}$ that allows us to learn the class of all functions that can be written as the pointwise maximum of k functions from $\mathcal{F}$. This allows us to exhibit a number of further nontrivial classes of functions from $\mathbb{R}$ into $\mathbb{R}$ for which there exist efficient learning algorithms.

Keywords: Computational learning theory, on-line learning, mistake-bounded learning, function learning

1. Introduction

We consider the complexity of function learning in the most common nonprobabilistic models of on-line learning. In contrast to the rather well-developed theory for the special case of $\{0,1\}$-valued functions (i.e., concepts), relatively little is known about general properties of optimal mistake bounds (resp. loss bounds) for learning classes of functions with larger ranges (e.g., real-valued functions). Furthermore, nontrivial positive learning results have so far been achieved in these models only for very few function classes.

The main learning model that we consider is the common model for on-line function learning. For some fixed class $\mathcal{F}$ of possible target functions T from X to Y the learner proposes at each round s of a learning process a hypothesis function $h_s : X \rightarrow Y$. The environment responds with a counterexample $\langle x, T(x) \rangle$ such that the learner's prediction $h_s(x)$ for argument x differs from the true value $T(x)$ of the target function $T(x)$. The loss of the learner at round s is measured by $\ell(h_s(x), T(x))$, for some suitable loss function $\ell : Y \times Y \rightarrow \mathbb{R}^+$ (e.g. $\ell(h_s(x), T(x)) = (h_s(x) - T(x))^2$ if $Y \subseteq \mathbb{R}$). The goal of the learner is to minimize his total loss, i.e. the sum of his losses for all rounds s. Hence one is interested in efficient learning algorithms for $\mathcal{F}$ that produce a suitable hypothesis h_s for each round s of any learning process so that the total loss of the learner for the worst case choice of $T \in \mathcal{F}$ and the worst case choice of counterexamples is as small as possible. A detailed definition of the resulting "learning complexity" for an arbitrary function class $\mathcal{F}$ (denoted by LC-ARB($\mathcal{F}$)) is given at the beginning of Section 2 of this paper.

This learning model is equivalent to a slightly different learning model, where the environment provides *arbitrary* examples (or "trials") $\langle x, T(x) \rangle$ of the target function $T : X \rightarrow Y$. These are not required to be "counterexamples" to the current hypothesis h_s of the learner (i.e. we may have $\ell(h_s(x), T(x)) = 0$). At round s the learner is given a point x_s and he makes a "prediction" $h_s(x_s)$, after which he receives the correct value $T(x_s)$. If $h_s(x_s) \neq T(x_s)$, one says that the learner makes a *mistake* at trial s (Littlestone, 1988). As before, the loss of the learner at round s is measured by $\ell(h_s(x), T(x))$, and his goal is to minimize the sum of his losses over all rounds. This variation of the learning model is a bit more plausible from the point of view of applications. It is a straightforward generalization of Littlestone's "mistake bounded" model for concept learning (Littlestone, 1988). However as in the case of concept learning (see (Littlestone, 1988)) one can also show very easily for function learning that this variation leads to the same definition of the learning complexity of a function class. Hence we prefer to work with the former version of the learning model, which is somewhat simpler to analyze.

Our model for function learning has already been widely studied, both theoretically and practically. For example the well-known back-propagation learning algorithm for function learning on neural nets is a learning algorithm for this type of learning model (however there one just wants to minimize the deviation of the hypothesis at the end of the training phase from the target function). In that special case the hypothesis h_s is the function that is computed by the neural net after s applications of the backwards propagation rule for adjusting the weights of the neural net (for s incorrectly processed training examples).

From the theoretical point of view this function learning model is a straightforward generalization of the common model for on-line concept learning in a worst case setting, since concepts may be viewed as functions with range $\{0, 1\}$ (see (Barzdin and Frievald, 1972), (Angluin, 1988), (Littlestone, 1988), (Maass and Turán, 1992)). In ((Dawid, 1984), (Mycielski, 1988) , (Vovk, 1990), (Littlestone, Long and Warmuth, 1991), (Faber and Mycielski, 1991), (Littlestone and Warmuth, 1991), (Kimber and Long, 1992), (Feder, Merhav and Gutman, 1992), (Vovk, 1992), (Cesa-Bianchi, Long and Warmuth, 1993)), the learning complexity of various concrete function classes have been investigated in this function learning model, and cumulative loss bounds in terms of certain properties of $\mathcal{F}$ have been proved for general-purpose (but not necessarily computationally efficient) algorithms ((Vovk, 1990), (Littlestone and Warmuth, 1991), (Feder, Merhav and Gutman, 1992), (Vovk, 1992)). Nevertheless the only interesting class of functions $f : \mathbb{R}^d \to \mathbb{R}$ for $d \geq 2$ that has been shown to be efficiently learnable is the class of linear functions ((Mycielski, 1988), (Littlestone, Long and Warmuth, 1991), (Cesa-Bianchi, Long and Warmuth, 1993)). Coming up with an efficient algorithm for linear functions with a decent loss bound is trivial, but coming up with optimal or near optimal loss bounded algorithms can be difficult.

In some learning settings, arising for example in control problems, the value $T(x)$ is not made available to the learner, and only weaker reinforcement is provided by the environment. At the end of Section 2, we consider variations of the previously described function learning model which model two such situations which arise in practice. In the first variation we assume that the learner only receives the information that $T(x) \neq h_s(x)$. In the second variation (studied previously by Barland (1992)) we assume that the learner is in addition told whether $T(x) > h_s(x)$ ("$h_s(x)$ is too low"), or whether $T(x) < h_s(x)$ ("$h_s(x)$ is too high").

In Section 2, we introduce a notion of an adversary tree (or mistake tree) for function learning. This notion provides an independent "dual" definition of the learning complexity of a function class. It turns out to be sufficient to consider *binary* branching adversary trees, even for learning *real-valued* functions under a large class of smooth loss functions including the heavily studied quadratic loss and "absolute loss." We apply this result to prove optimal lower bounds, in terms of LC-ARB($\mathcal{F}$), both for probabilistic function learning algorithms in an oblivious environment and for function learning with a wide variety of queries. At the end of Section 2, we exhibit appropriate notions of an adversary tree for two other function learning models.

In Section 3, we derive general upper bounds on the number LC-ARB($\cup_{i=1}^{p}\mathcal{F}_i$) of mistakes required for learning $\cup_{i=1}^{p}\mathcal{F}_i$ in terms of LC-ARB($\mathcal{F}_1$), ... ,LC-ARB($\mathcal{F}_p$). These bounds are shown to be tight to within a constant factor. The results of Section 3 remain true, and are new, even for the special case of $\{0,1\}$-valued functions (i.e., concepts).

In Section 4, we exhibit two algorithms for learning the maximum of k linear functions over $\mathbb{R}^d$. The first has a mistake bound polynomial in k for fixed d, and the second has a mistake bound polynomial in d for fixed k. To the best of our knowledge, these provide the first positive learning results in the LC-ARB model for any nontrivial class of functions defined on $\mathbb{R}^2$ other than linear functions.

In Section 5, we exhibit a more general algorithm for learning the maximum of k real-valued functions of a single real variable. It yields as special cases algorithms which have mistake bounds linear in k for learning the maximum of k polynomials (with a bounded number of terms), of k rational functions (again, with a bounded number of terms), and of k sigmoidal functions.

This paper provides full proofs and more detailed explanations of the results described in the previously published extended abstract (Auer, Long, Maass and Woeginger, 1993).

2. Adversary Trees for Function Learning

We introduce in Definition 2 our new notion of an adversary tree for function learning, which allows us to give in Theorem 1 a completely independent ("dual") definition of the learning complexity of a function class. A somewhat surprising feature of Definition 2 is that it suffices for Theorem 1 if we restrict our attention to *binary* branching trees, independently of the size of the range Y of the functions that are learned. This allows us to derive in Theorem 2 an *optimal* lower bound for randomized algorithms for function learning in an oblivious environment. As another consequence of Theorem 1 we derive in Theorem 3 an optimal lower bound for function learning with a large variety of queries.

Assume that X ("domain"), Y ("range"), and A ("action space ") are some arbitrary sets (finite or infinite).[1] Consider some arbitrary sets $\mathcal{F} \subseteq Y^X$ ("function class") and $\mathcal{H} \subseteq A^X$ ("hypothesis space"). Let $\ell : A \times Y \to \mathbb{R}^+ := \{r \in \mathbb{R} : r \geq 0\}$ be some arbitrary function ("loss function"). Note that we consider a slightly more general setting than outlined in Section 1, since we allow here for the possibility that the learner proposes for each $x \in X$ some "action" $h(x)$ that lies in some suitable "action space" A. In this general framework the loss function $\ell : A \times Y \to \mathbb{R}^+$ measures how bad the proposed action $h(x)$ is, with respect to the given reinforcement $y \in Y$.

A learning process is a dialogue between the "learner" and the "environment". At the beginning of each learning process the environment fixes some $T \in \mathcal{F}$ ("target function"). At each round s of the learning process ($s = 1, 2, \ldots$) the learner proposes some hypothesis $h_s \in \mathcal{H}$. The environment responds at the same round with a pair $\langle x_s, T(x_s)\rangle$ for some $x_s \in X$. The *loss* of the learner at round s is $\ell(h_s(x_s), T(x_s))$. If $h_s(x_s) \neq T(x_s)$ we also refer to $\langle x_s, T(x_s)\rangle$ as a *counterexample* (CE) to hypothesis h_s. The *total loss* of the learner for this learning process is $\sum_{s=1}^{t} \ell(h_s(x_s), T(x_s))$, where $t \in \mathbb{N} \cup \{\infty\}$ is the number of rounds in this learning process.

A *learning algorithm* B for $\mathcal{F}$ with hypothesis space $\mathcal{H}$ is a function which produces for any $T \in \mathcal{F}$ at each round s of such learning process some hypothesis

$$h_s = B(\langle x_1, T(x_1)\rangle, \ldots, \langle x_{s-1}, T(x_{s-1})\rangle; h_0, \ldots, h_{s-1}) \in \mathcal{H}$$

of the learner. We will consider in this paper only deterministic learning algorithms, for which $h_0, \ldots, h_{s-1}$ are not actually needed as arguments for B in the definition of

h_s (since they can be recomputed with the help of the preceding counterexamples). We define $\mathrm{LC}_\ell(B)$ as the supremum of the total losses of the learner for learning processes with learning algorithm B and loss function ℓ. We set

$$\begin{aligned}\mathrm{LC}_\ell(\mathcal{F},\mathcal{H}) := \inf\{&\mathrm{LC}_\ell(B) : B \text{ is a deterministic}\\ &\text{learning algorithm for } \mathcal{F} \text{ with hypothesis space } \mathcal{H}\}\end{aligned}$$

and $\mathrm{LC\text{-}ARB}_\ell(\mathcal{F}) := \mathrm{LC}_\ell(\mathcal{F}, A^X)$. These definitions contain as special cases those that were considered in ((Mycielski, 1988), (Littlestone and Warmuth, 1991), (Littlestone, Long and Warmuth, 1991), (Faber and Mycielski, 1991), (Kimber and Long, 1992)) for functions, and in ((Angluin, 1988), (Littlestone, 1988), (Maass and Turán, 1992)) for concepts.

Starting in Application 2 of this section, we will consider only the discrete loss function ℓ defined by $\ell(a,y) = 0$ if $a = y$, and $\ell(a,y) = 1$ if $a \neq y$. From that point on, we will drop the subscript ℓ in $\mathrm{LC\text{-}ARB}_\ell$.

Remark.

a) One can easily see that the size of $\mathrm{LC}_\ell(\mathcal{F},\mathcal{H})$ does not change if we only allow responses $\langle x_s, T(x_s)\rangle$ with $\ell(h(x_s), T(x_s)) > 0$ in a learning process. This will be assumed w.l.o.g. throughout this paper (except for Theorem 2, which concerns another model).

b) In contrast to the discrete case there does not always exist a learning algorithm B for $\mathcal{F}$ with $\mathrm{LC}_\ell(B) = \mathrm{LC\text{-}ARB}_\ell(\mathcal{F})$ (i.e., the infimum of the $\mathrm{LC}_\ell(B)$ need not be obtained). An easy example is constructed by setting $X = A = Y = (0,1]$, $\mathcal{F} = \{f : X \to Y : f \text{ is constant}\}$, and $\ell(a,y) = 0$ if $y \geq a$, else $\ell(a,y) = a - y$. Then for any learning algorithm B one has $\mathrm{LC}_\ell(B) = h(1) \in (0,1]$, where h is the first hypothesis of B.

We will show in the subsequent Theorem 1 that for learning real valued functions under any of the common loss functions one can assume without loss of generality that the "adversary" (i.e. the environment) is very nice to the learner. One can assume that at the *beginning* of each round s in a learning process (*before* the learner has issued his hypothesis h_s for this round) the adversary tells the learner which point x_s he is going to use for his counterexample $\langle x_s, y\rangle$ at this round. In addition the adversary also gives the learner a set $\{y_1, y_2\} \subseteq Y$ with the guarantee that the second component y of the counterexample $\langle x_s, y\rangle$ will be an element of $\{y_1, y_2\}$. Finally the adversary also announces to the learner the rule by which he will choose y_1 or y_2 (depending on $h_s(x_s)$): he gives to the learner a set A_1 such that $y = y_1 \Leftrightarrow h_s(x_s) \in A_1$.

Our intuition tells us that it is essential for an "optimal" adversary that he chooses x_s *after* he has seen h_s (as a "weak spot of h_s"), and that he exploits the full size of the range Y to choose a reinforcement y that causes a largest possible loss $\ell(h_s(x_s), y)$ to the learner. The subsequent Theorem 1 tells us that this intuition is wrong.

Whereas a general adversary strategy is a rather complex mathematical object (since in general each move of the adversary depends on the preceding hypotheses of the learner), any adversary strategy of the previously described simple type can easily be described by a binary branching tree according to the following definition.

Definition. An adversary tree $\mathcal{U}$ for a function class $\mathcal{F} \subseteq Y^X$ and an action set A is a finite binary branching tree whose nodes and edges are labeled in the following way:

Each node ν of $\mathcal{U}$ is labeled by a pair $\langle x_\nu, \mathcal{F}_\nu \rangle$ with $x_\nu \in X$, $\mathcal{F}_\nu \subseteq \mathcal{F}$ and $\mathcal{F}_\nu \neq \emptyset$. If ν is the root of $\mathcal{U}$ we set $\mathcal{F}_\nu := \mathcal{F}$.

If ν is not a leaf of $\mathcal{U}$, then its two outgoing edges have labels of the form $\langle A_1, y_1 \rangle$ and $\langle A_2, y_2 \rangle$, where A_1, A_2 is a nontrivial partition of A (i.e. $A_1 \cup A_2 = A$, $A_1 \cap A_2 = \emptyset$, $A_1 \neq \emptyset$ and $A_2 \neq \emptyset$) and y_1, y_2 are elements of Y such that $y_1 \neq y_2$ and $\{f \in \mathcal{F}_\nu : f(x_\nu) = y_i\} \neq \emptyset$ for $i = 1, 2$. The set $\{f \in \mathcal{F}_\nu : f(x_\nu) = y_i\}$ is then the second component of the label of the node at which the edge with label $\langle A_i, y_i \rangle$ ends $(i = 1, 2)$.

For any loss function $\ell : A \times Y \to \mathbb{R}^+$, one defines the loss $\ell(\tilde{A}, y)$ of an edge in $\mathcal{U}$ with label $\langle \tilde{A}, y \rangle$ by $\ell(\tilde{A}, y) := \inf\{\ell(a, y) : a \in \tilde{A}\}$. The total loss of a path in $\mathcal{U}$ is the sum of the losses of edges in this path.

We set

$$\begin{aligned} \mathrm{INF}_\ell(\mathcal{U}) := \quad & \inf\{K \in \mathbb{R}^+ : K \text{ is the total loss of a path in } \mathcal{U} \text{ that} \\ & \text{starts at the root and ends at a leaf of } \mathcal{U}\} \text{ and} \\[1em] \mathrm{ADV}_\ell(\mathcal{F}) := \quad & \sup\{\mathrm{INF}_\ell(\mathcal{U}) : \mathcal{U} \text{ is an adversary tree for} \\ & \mathcal{F} \text{ with action space } A \text{ and loss function } \ell\}. \end{aligned}$$

Obviously any adversary tree $\mathcal{U}$ for $\mathcal{F}$ encodes an adversary strategy which forces the learner to take a total loss of at least $\mathrm{INF}_\ell(\mathcal{U})$, no matter which hypotheses he chooses in the course of the learning process. The second part $\mathcal{F}_\nu$ of the label $\langle x_\nu, \mathcal{F}_\nu \rangle$ of an internal node ν of $\mathcal{U}$ specifies a set of functions which could still be choosen as target functions by the adversary. The labels $\langle A_1, y_1 \rangle$ and $\langle A_2, y_2 \rangle$ of the two edges that leave the node x_ν encode the following rule for the adversary at the next step of a learning process for $\mathcal{F}$: If $h(x_0) \in A_i$ (where h is the next hypothesis of the learner) then the adversary responds with the pair $\langle x_0, y_i \rangle$ (which forces the learner to take a loss of at least $\ell(A_i, y_i)$ at this step).

One also can easily read off from the definition of an adversary tree $\mathcal{U}$ that it only encodes "particularly nice" adversary strategies, since one may assume that the learner has full knowledge of $\mathcal{U}$. In particular at each step of a learning process where the adversary follows the strategy encoded by $\mathcal{U}$, the first component x_0 and the rule by which the adversary chooses the second component y_i of his next counterexample $\langle x_0, y_i \rangle$ are already known to the learner *before* he chooses his hypothesis h for this step.

Figure 1. Example of an adversary tree $\mathcal{U}$ for $X = \{a, b, c\}$, $A = Y = \{1, 2, 3\}$ and $\mathcal{F} = Y^X$ (where the first components of the labels of the leafs have been deleted since they are irrelevant). For the loss function ℓ with $\ell(a, y) = |a - y|$ one has $\mathrm{INF}_\ell(\mathcal{U}) = 2$.

Intuitively one may tend to believe that $\text{LC-ARB}_\ell(\mathcal{F}) > \mathrm{ADV}_\ell(\mathcal{F})$ for many function classes $\mathcal{F}$, i.e. that the environment becomes significantly weaker if it limits its adversary strategies to those particularly simple ones that can be encoded by adversary trees. The following theorem asserts that this is *not* the case. The assumptions of this theorem contain a rather trivial condition (+) which is explained and motivated in the subsequent Remark .

THEOREM 1 *Let X, Y, A, and $\mathcal{F} \subseteq Y^X$ be arbitrary nonempty sets (finite or infinite) and let $\ell : A \times Y \to \mathbb{R}^+$ be a function such that*

$$(+) \quad \forall \rho > 0 \; \forall x \in X \; \forall f \in \mathcal{F} \; \exists a \in A \; (\ell(a, f(x)) \leq \rho),$$

and

(i) *$A = Y$ and ℓ is the "discrete loss function" with $\ell(a, y) = 0$ if $a = y$ and $\ell(a, y) = 1$ if $a \neq y$, or*

(ii) *A and Y are subsets of $\mathbb{R}$ and the loss function ℓ is of the form $\ell(a, y) = \mathrm{L}(|a - y|)$ for some arbitrary nondecreasing and continuous function $\mathrm{L} : \mathbb{R}^+ \to \mathbb{R}^+$ with $\mathrm{L}(0) = 0$ (e.g. $\ell(a, y) = |a - y|^p$ for some arbitrary parameter $p > 0$).*

Then $\text{LC-ARB}_\ell(\mathcal{F}) = \mathrm{ADV}_\ell(\mathcal{F})$.

Remark. The assumptions of the preceding theorem contain the technical condition (+), which just says that the set A of possible outputs ("actions") of the learner is sufficiently large. More precisely, it demands that for any possible value $f(x)$ of a target function $f \in \mathcal{F}$ there exist points $a \in A$ which make the loss $\ell(a, f(x))$ of the learner for argument x arbitrarily small. This condition is hardly restrictive, since usually one even has $A = Y$ (e.g. $A = Y = [0,1]$). Furthermore it is obvious that $\text{LC-ARB}_\ell(\mathcal{F}) = \infty$ if condition (+) is not satisfied. Thus (+) holds for all learning problems that are of interest in this context.

We also would like to point out that (+) is a necessary assumption for Theorem 1. This follows by considering an example where $\mathcal{F}$ consists of a single function $f : \mathbb{R} \to \mathbb{R}$ (hence $\text{ADV}_\ell(\mathcal{F}) = 0$), and $A \subseteq \mathbb{R}$ is chosen such that $\inf\{\ell(a, f(x_0)) : a \in A\} > 0$ for some $x_0 \in \mathbb{R}$ (hence (+) is not satisfied, thus $\text{LC-ARB}_\ell(\mathcal{F}) = \infty$).

Proof of Theorem 1: We start by showing that $\text{LC-ARB}_\ell(\mathcal{F}) \leq \text{ADV}_\ell(\mathcal{F})$ in the **case (ii)**. Fix some arbitrary $\varepsilon > 0$. In order to show that $\text{LC-ARB}_\ell(\mathcal{F}) \leq \text{ADV}_\ell(\mathcal{F}) + \varepsilon$ we choose an arbitrary sequence $(\varepsilon_j)_{j \in \mathbb{N}}$ of reals $\varepsilon_j > 0$ such that $\varepsilon = \sum\limits_{j=1}^{\infty} \varepsilon_j$. It suffices to construct a learning algorithm B so that for every $s \in \mathbb{N}$ the following property holds (which implies that $\text{LC}_\ell(B) \leq \text{ADV}_\ell(\mathcal{F}) + \varepsilon$):

(*) For every learning process with learning algorithm B and examples

$$\langle x_1, y_1 \rangle, \ldots, \langle x_s, y_s \rangle$$

the proposed actions $a_1, \ldots, a_s$ of B (where $a_j := h_j(x_j)$ for the hypothesis h_j of B at round j) satisfy

$$\text{ADV}_\ell(\{f \in \mathcal{F} : f(x_j) = y_j \text{ for } j = 1, \ldots, s\}) + \sum_{j=1}^{s} \ell(a_j, y_j)$$
$$\leq \text{ADV}_\ell(\mathcal{F}) + \sum_{j=1}^{s} \varepsilon_j .$$

The property (*) is trivially satisfied for $s = 0$. Assume now that (*) holds for some arbitrary $s \in \mathbb{N}$. In order to prove (*) for $s+1$ we fix some learning process with learning algorithm B and set $\tilde{\mathcal{F}} := \{f \in \mathcal{F} : f(x_j) = y_j \text{ for } j = 1, \ldots, s\}$. The hypothesis h of B for the next round $s+1$ of the learning process is defined as follows:

For every $x \in X$ let $h(x)$ be some $a \in A$ such that

$$\text{ADV}_\ell(\{f \in \tilde{\mathcal{F}} : f(x) = y\}) + \ell(a, y) \leq \text{ADV}_\ell(\tilde{\mathcal{F}}) + \varepsilon_{s+1}$$

for all $y \in Y$ for which there exists an $f \in \tilde{\mathcal{F}}$ such that $f(x) = y$.

It is obvious that if the learning algorithm B chooses as hypothesis h_{s+1} at step $s+1$ a hypothesis h with the preceding property, then (*) is also satisfied for $s+1$. Hence it just remains to show that there exists a hypothesis h with the preceding property. This proof turns out to be rather nontrivial, involving subtle arguments from real valued analysis.

LEMMA 1 *The previously defined function $h : X \to A$ is well-defined for all $x \in X$.*

Proof of Lemma 1: Assume for a contradiction that $h(x_0)$ is not well-defined for some $x_0 \in X$, i.e.

$$
\begin{aligned}
(^{**})\quad & \forall a \in A \quad \exists y(a) \in Y(\exists f \in \tilde{\mathcal{F}}(f(x_0) = y(a)) \\
& \wedge \mathrm{ADV}_\ell(\{f \in \tilde{\mathcal{F}} : f(x_0) = y(a)\}) + \ell(a, y(a)) \ > \ \mathrm{ADV}_\ell(\tilde{\mathcal{F}}) + \varepsilon_{s+1}).
\end{aligned}
$$

Intuitively this assumption (**) means that for every proposed action $a = h(x_0)$ of the learner for the fixed argument x_0 there is a legal countermove of the adversary (where he gives a value $y(a)$ as the correct value $f(x_0)$ of the target function f at argument x_0) which causes an "unusually large" loss to the learner. In this context the loss $\ell(a, y(a))$ is "unusually large" if the sum of $\ell(a, y(a))$ and of the remaining "adversary power" $\mathrm{ADV}_\ell(\{f \in \tilde{\mathcal{F}} : f(x_0) = y(a)\})$ after this countermove exceeds the "adversary power" $\mathrm{ADV}_\ell(\tilde{\mathcal{F}})$ at the beginning of this move by more then the previously specified nonzero quantity ε_{s+1}.

The only chance to refute this assumption (**) is to show that then we could *select two of these countermoves* ($y(a_1)$ and $y(a_2)$ for two suitable actions a_1, a_2) and a partition of the action space A into subsets A_1 and A_2 such that the learner would still suffer an "unusually large" (although possibly by a fraction of ε_{s+1} smaller) loss if the adversary would for $i = 1$ and $i = 2$ always respond with the *same* countermove $y(a_i)$ for any proposed action $a = h(x_0) \in A_i$. This would lead to a contradiction to the definition of $\mathrm{ADV}_\ell(\tilde{\mathcal{F}})$ (as the smallest total loss along a root-leaf path in a tree where the smallest total loss along any root-leaf path is as large as possible), since we could then construct an adversary tree $\mathcal{U}$ for $\tilde{\mathcal{F}}$ which has $\langle x_0, \tilde{\mathcal{F}}\rangle$ as label of the root, $\langle A_1, y(a_1)\rangle$ and $\langle A_2, y(a_2)\rangle$ as labels of the edges from the root, and an almost optimal adversary tree $\mathcal{U}_i$ for the subclass $\{f \in \tilde{\mathcal{F}} : f(x_0) = y(a_i)\}$ attached below the edge with label $\langle A_i, y(a_i)\rangle$ (for $i = 1, 2$). This adversary tree $\mathcal{U}$ would then have on any root-leaf path a total loss that exceeds $\mathrm{ADV}_\ell(\tilde{\mathcal{F}})$ by $\frac{\varepsilon_{s+1}}{2}$, contradicting the definition of $\mathrm{ADV}_\ell(\tilde{\mathcal{F}})$.

In order to carry out this plan for refuting (**) one has to look at the concrete structure of any possible function $a \mapsto y(a)$ that satisfies the conditions of (**). For example if there exist some $a_1, a_2 \in A$ such that $y(a_1) \leq a_1 < a_2 \leq y(a_2)$ (this is Case 1 in the subsequent analysis), then one can easily read off from Figure 2 that the assumed monotonicity of the function L with $\ell(a, y) = \mathrm{L}(|a - y|)$ implies that for $A_1 := \{a \in A : a \geq a_1\}$ and $A_2 := A - A_1$ the desired proporties are met (since $\ell(A_i, y(a_i)) \geq \ell(a_i, y_i)$ for $i = 1, 2$).

If there are no actions a_1, a_2 as above, then we may conclude that the set $\{a \in A : y(a) \geq a\}$ is "closed to the left", i.e. $\forall a, a' \in A((y(a) \geq a \wedge a' < a) \Rightarrow y(a') \geq a')$ (this is Case 2 of the subsequent precise proof). In this case one either has $\forall a \in A(y(a) \geq a)$ (this is Case 2.1), or A can be partitioned into a left interval $\{a \in A : y(a) \geq a\}$ with supremum s_1 and a right interval $\{a \in A : y(a) < a\}$

Figure 2. Definition of A_1 and A_2 in Case 1.

with infinum $s_2 \geq s_1$ (this is Case 2.2). The idea for the definition of a_1, a_2, A_1, A_2 for these two subcases can be easily read off from Figure 3 respectively Figure 4 below. The precise construction is a bit more involved, since it also depends on the concrete structure of the loss function ℓ. However the reader may skip the subsequent detailed proof without loss of understanding for the later results in this paper.

The precise refutation of assumption (**) proceeds as follows. We fix for each $a \in A$ some $y(a) \in Y$ as in (**). In each of the following cases we get a contradiction by building an adversary tree $\mathcal{U}$ for $\tilde{\mathcal{F}}$ which satisfies $\mathrm{INF}_\ell(\mathcal{U}) \geq \mathrm{ADV}_\ell(\tilde{\mathcal{F}}) + \frac{\varepsilon_{s+1}}{2}$ (hence $\mathcal{U}$ provides a contradiction to the definition of $ADV_\ell(\tilde{\mathcal{F}})$). The label of the root of $\mathcal{U}$ will always be $\langle x_0, \tilde{\mathcal{F}} \rangle$.

Case 1: $\boldsymbol{\exists a_1, a_2 \in A \quad (y(a_1) \leq a_1 < a_2 \leq y(a_2))}$.
Then $\ell(a, y(a_1)) \geq \ell(a_1, y(a_1))$ for all $a \geq a_1$, and $\ell(a, y(a_2)) \geq \ell(a_2, y(a_2))$ for all $a \leq a_2$. We set $A_1 := \{a \in A : a \geq a_1\}$ and $A_2 := A - A_1$.

We choose $\langle A_1, y(a_1) \rangle$ and $\langle A_2, y(a_2) \rangle$ as labels of the edges that leave the root of the constructed adversary tree $\mathcal{U}$. Below the edge with label $\langle A_i, y(a_i) \rangle$ we attach an adversary tree $\mathcal{U}_i$ for $\{f \in \tilde{\mathcal{F}} : f(x_0) = y(a_i)\}$ that satisfies $\mathrm{INF}_\ell(\mathcal{U}_i) \geq \mathrm{ADV}_\ell(\{f \in \tilde{\mathcal{F}} : f(x_0) = y(a_i)\}) - \frac{\varepsilon_{s+1}}{2}$, for $i = 1, 2$. By definition of A_i we have $\ell(A_i, y(a_i)) \geq \ell(a_i, y(a_i))$ for $i = 1, 2$. Hence

$$\begin{aligned} \mathrm{INF}_\ell(\mathcal{U}) &= \min\{\mathrm{INF}_\ell(\mathcal{U}_i) + \ell(A_i, y(a_i)) : i = 1, 2\} \\ &\geq \min\{\mathrm{ADV}_\ell(\{f \in \tilde{\mathcal{F}} : f(x_0) = y(a_i)\}) - \tfrac{\varepsilon_{s+1}}{2} + \ell(A_i, y(a_i)) : i = 1, 2\} \\ &\geq \mathrm{ADV}_\ell(\tilde{\mathcal{F}}) + \tfrac{\varepsilon_{s+1}}{2} \end{aligned}$$

by the definition of $y(a_i)$ for $i = 1, 2$.

Case 2: $\boldsymbol{\forall a, a' \in A \quad ((y(a) \geq a \wedge a' < a) \Rightarrow y(a') \geq a')}$.
We define

Figure 3. Definition of A_1 and A_2 in Case 2.1

$$s_1 := \sup\{a \in A : y(a) \geq a\} \quad \text{and} \quad s_2 := \inf\{a \in A : y(a) < a\}$$

(as usually, we set $\sup \emptyset := -\infty$ and $\inf \emptyset := \infty$). The assumption of this case implies that $s_1 \leq s_2$.

Case 2.1: $\forall a \in A \quad (y(a) \geq a)$.
We first show that $s_1 = \infty$ in this subcase. Assume for a contradiction that $s_1 < \infty$. Set

$$z_0 := \inf\{z > 0 : \mathrm{L}(z) > 0\}.$$

Then $\mathrm{L}(z_0) = 0$. Since $\ell(a, y(a)) \geq \varepsilon_{s+1}$ for all $a \in A$, there exists some $\delta > 0$ such that $y(a) > s_1$ for all $a \in A$ with $|a - s_1| \leq \delta$. If $|s_1 - y(a)| \leq z_0$ for all these $a \in A$, then this would imply that $\ell(a, y(a)) \to 0$ for $a \to s_1$. If $|s_1 - y(a_1)| > z_0$ for some $a_1 \in A$ with $y(a_1) > s_1$, then $\mathrm{L}(|a - y(a_1)|) \geq \mathrm{L}(|s_1 - y(a_1)|) > 0$ for all $a \in A$. This provides a contradiction to the assumption (+) of Theorem 1. Hence we may assume that $s_1 = \infty$.

Fix any $a_1 \in A$. Choose $a_2 \in A$ such that $a_2 > y(a_1)$ and $\ell(a_2, y(a_1)) \geq \ell(a_1, y(a_1))$. Set $A_1 := \{a \in A : a > a_2\}$ and $A_2 := \{a \in A : a \leq a_2\}$ (see Figure 2). We then have $\ell(A_1, y(a_1)) \geq \ell(a_1, y(a_1))$ and $\ell(A_2, y(a_2)) \geq \ell(a_2, y(a_2))$. Construct $\mathcal{U}$ analogously as in Case 1.

Case 2.2: $\exists a_1, a_2 \in A \quad (y(a_1) \geq a_1 \wedge y(a_2) < a_2)$.
We then have $-\infty < s_1 \leq s_2 < \infty$ by the definition of Case 2. Our strategy in this subcase is to choose $a_1 < s_1$ so close to s_1 that $y(a_1) > s_1$, and that $\ell(s_1, y(a_1))$ differs from $\ell(a_1, y(a_1))$ by at most $\frac{\varepsilon_{s+1}}{4}$. Analogously we want to choose $a_2 > s_2$ so close to s_2 that $y(a_2) < s_2$ and that $\ell(s_2, y(a_2))$ differs from $\ell(a_2, y(a_2))$ by at most $\frac{\varepsilon_{s+1}}{4}$ (see Figure 4). It turns out that such choice of a_1, a_2 is always possible except for subcase 2.2.2, where we trivially succeed with a slightly different choice of a_1, a_2.

Figure 4. Definition of A_1 and A_2 in Case 2.2

Set

$$\begin{aligned} r_1 &:= \sup\{|a - y(a)| : a \in [s_1 - 1, s_1] \cap A\} \text{ and} \\ r_2 &:= \sup\{|a - y(a)| : a \in [s_2, s_2 + 1] \cap A\}. \end{aligned}$$

Case 2.2.1: $r_1 < \infty$ and $r_2 < \infty$.
Set $r := \max(r_1, r_2)$. Then the continuous function L is uniformly continuous on the compact interval $[0, r]$. Hence there exists some $\varepsilon' > 0$ (w.l.o.g. $\varepsilon' \le 1$) such that $|\mathrm{L}(z) - \mathrm{L}(z')| \le \frac{\varepsilon_{s+1}}{4}$ for any $z, z' \in [0, r]$ with $|z - z'| \le \varepsilon'$. Define as in Case 2.1 $z_0 := \inf\{z \ge 0 : \mathrm{L}(z) > 0\}$.

Define $A_1 := \{a \in A : y(a) \ge a\}$ and $A_2 := \{a \in A : y(a) < a\}$. We want to choose $a_1 \in A$ such that $|a_1 - s_1| \le \varepsilon'$, $\mathrm{L}(|a_1 - s_1|) \le \frac{\varepsilon_{s+1}}{2}$, and $\mathrm{L}(|s_1 - y(a_1)|) > 0$. If $|s_1 - y(a)| \le z_0$ for all $a \in A_1$ with $|a - s_1| \le \varepsilon'$ and $\mathrm{L}(|a - s_1|) \le \frac{\varepsilon_{s+1}}{2}$, then $\mathrm{L}(|a - y(a)|) \to 0$ for $a \to s_1$, $a \in A_1$ (since $||a - y(a)| - |s_1 - y(a)|| \to 0$ for $a \to s_1$, $a \in A_1$). This is a contradiction to the fact that $\mathrm{L}(|a - y(a)|) \ge \varepsilon_{s+1}$ for all $a \in A$ (by the definition of $y(a)$). Hence there exists some $a_1 \in A_1$ with $|a_1 - s_1| \le \varepsilon'$, $\mathrm{L}(|a_1 - s_1|) \le \frac{\varepsilon_{s+1}}{2}$, and $|s_1 - y(a_1)| > z_0$ (thus $\mathrm{L}(|s_1 - y(a_1)|) > 0$). Analogously there exists some $a_2 \in A_2$ with $|a_2 - s_2| \le \varepsilon'$, $\mathrm{L}(|a_2 - s_2|) \le \frac{\varepsilon_{s+1}}{2}$, and $\mathrm{L}(|s_2 - y(a_2)|) > 0$.

Since $\mathrm{L}(|a_i - y(a_i)|) \ge \varepsilon_{s+1}$ by the definition of $y(a_i)$, and since $\mathrm{L}(|a_i - s_i|) \le \frac{\varepsilon_{s+1}}{2}$ by the choice of a_i, we have $y(a_1) > s_1$ and $y(a_2) < s_2$. Furthermore, we can conclude that $y(a_1) \ne y(a_2)$, since otherwise $s_1 < y(a_1) = y(a_2) < s_2$. This would provide a contradiction to (+) because $(s_1, s_2) \cap A = \emptyset$ by the definition of s_1, s_2, and $\mathrm{L}(|s_i - y(a_i)|) > 0$ by the choice of a_i for $i = 1, 2$.

We have $|a_i - y(a_i)| \le r$ since $\varepsilon' \le 1$. Hence $\ell(A_i, y(a_i)) = \inf\{\mathrm{L}(|a - y(a_i)|) : a \in A_i\} \ge \mathrm{L}(|s_i - y(a_i)|) \ge \mathrm{L}(|a_i - y(a_i)|) - \frac{\varepsilon_{s+1}}{4} = \ell(a_i, y(a_i)) - \frac{\varepsilon_{s+1}}{4}$, since $|a_i - y(a_i)|$, $|s_i - y(a_i)| \in [0, r]$ and $||a_i - y(a_i)| - |s_i - y(a_i)|| \le |a_i - s_i| \le \varepsilon'$, $i = 1, 2$. Therefore we can construct an adversary tree $\mathcal{U}$ for $\tilde{\mathcal{F}}$ with $\mathrm{INF}_\ell(\mathcal{U}) \ge \mathrm{ADV}_\ell(\tilde{\mathcal{F}}) + \frac{\varepsilon_{s+1}}{2}$ by attaching below the edge with label $\langle A_i, y(a_i)\rangle$ an adversary tree $\mathcal{U}_i$ for $\{f \in \tilde{\mathcal{F}} : f(x_0) = y(a_i)\}$ with $\mathrm{INF}_\ell(\mathcal{U}_i) \ge \mathrm{ADV}_\ell(\{f \in \tilde{\mathcal{F}} : f(x_0) = y(a_i)\} - \frac{\varepsilon_{s+1}}{4}$, $i = 1, 2$.

Case 2.2.2: $r_1 = r_2 = \infty$.
If $\sup\{\mathrm{L}(z) : z \in \mathrm{I\!R}^+\} < \infty$, then L is uniformly continuous on $\mathrm{I\!R}^+$ (since L is nondecreasing). Hence we can argue as in Case 2.2.1, with $\mathrm{I\!R}^+$ instead of $[0, r]$.

We assume in the following that $\sup\{\mathrm{L}(z) : z \in \mathrm{I\!R}^+\} = \infty$. Define $A_1 := \{a \in A : y(a) \geq a\}$ and $A_2 := \{a \in A : y(a) < a\}$. We can then conclude that $\sup\{\ell(A_1, y(a)) : a \in [s_1 - 1, s_1] \cap A\} = \infty$ and $\sup\{\ell(A_2, y(a)) : a \in [s_2, s_2 + 1] \cap A\} = \infty$. Hence it suffices for the construction of $\mathcal{U}$ to choose some $a_1 \in [s_1 - 1, s_1] \cap A$ such that $\ell(A_1, y(a_1)) \geq \mathrm{ADV}_\ell(\tilde{\mathcal{F}}) + \varepsilon_{s+1}$, and some $a_2 \in [s_2, s_2+1] \cap A$ such that $\ell(A_2, y(a_2)) \geq \mathrm{ADV}_\ell(\tilde{\mathcal{F}}) + \varepsilon_{s+1}$. The resulting adversary tree $\mathcal{U}$ for $\tilde{\mathcal{F}}$ of depth 1 satisfies $\mathrm{INF}_\ell(\mathcal{U}) \geq \mathrm{ADV}_\ell(\tilde{\mathcal{F}}) + \varepsilon_{s+1}$.

Case 2.2.3: $|\{i \in \{1, 2\} : r_i < \infty\}| = 1$.
In this case we use for i with $r_i < \infty$ a construction of $\langle A_i, y_i \rangle$ and $\mathcal{U}_i$ as in Case 2.2.1, and for $j \in \{1, 2\} - \{i\}$ a construction of $\langle A_j, y_j \rangle$ (and $\mathcal{U}_j$) as in Case 2.2.2.

Case 2.3: $\forall a \in A \quad (y(a) < a)$.
This case is dual to Case 2.1, and can be handled analogously. ■

This completes the proof of Lemma 1, and the proof that

$$\text{LC-ARB}_\ell(\mathcal{F}) \leq \mathrm{ADV}_\ell(\mathcal{F})$$

in case (ii).

The inequality $\text{LC-ARB}_\ell(\mathcal{F}) \geq \mathrm{ADV}_\ell(\mathcal{F})$ is shown in case (ii) as follows. Assume for a contradiction that $\text{LC-ARB}_\ell(\mathcal{F}) < \mathrm{ADV}_\ell(\mathcal{F})$. Fix some adversary tree $\mathcal{U}$ for the considered $\mathcal{F}$ and A such that $\mathrm{INF}_\ell(\mathcal{U}) \geq \text{LC-ARB}_\ell(\mathcal{F}) + \rho$ for some $\rho > 0$. Let B be an arbitrary learning algorithm for $\mathcal{F}$ with hypothesis space A^X. The adversary tree $\mathcal{U}$ provides in the obvious way responses to the hypotheses h of B, until a leaf of $\mathcal{U}$ has been reached. If one has arrived in $\mathcal{U}$ at a node ν with label $\langle x_\nu, \mathcal{F}_\nu \rangle$, $\langle A_1, y_1 \rangle$ and $\langle A_2, y_2 \rangle$ are the labels of the two outgoing edges, and $h(x_\nu) \in A_i$ $(i \in \{1, 2\})$ for the current hypothesis $h \in A^X$ of learning algorithm B, then the adversary gives $\langle x_\nu, y_i \rangle$ as his next response. Note that this response is a valid move of the adversary, since by definition of $\mathcal{U}$ there exists some $f \in \mathcal{F}_\nu$ with $f(x_\nu) = y_i$. The loss $\ell(h(x_\nu), y_i)$ of the learner at this round satisfies $\ell(h(x_\nu), y_i) \geq \ell(A_i, y_i)$. The adversary then proceeds in $\mathcal{U}$ to the node at the end of the edge with label $\langle A_i, y_i \rangle$. In this way the learning algorithm B defines a path from the root to some leaf of $\mathcal{U}$. By definition of $\mathrm{INF}_\ell(\mathcal{U})$ the total loss K of the learner during the resulting learning process satisfies $K \geq \mathrm{INF}_\ell(\mathcal{U})$. This implies that $K \geq \text{LC-ARB}_\ell(\mathcal{F}) + \rho$. Since B was chosen arbitrarily, this inequality provides a contradiction to the definition of $\text{LC-ARB}_\ell(\mathcal{F})$.

For the proof of Theorem 1 in **case (i)** we first note that the inequality

$$\text{LC-ARB}_\ell(\mathcal{F}) \geq \mathrm{ADV}_\ell(\mathcal{F})$$

can be shown in the same way as for case (ii). However this proof is simpler since $\text{LC-ARB}_\ell(\mathcal{F})$, $\text{ADV}_\ell(\mathcal{F}) \in \mathbb{N} \cup \{\infty\}$ for the discrete loss function ℓ. The inequality $\text{LC-ARB}_\ell(\mathcal{F}) \leq \text{ADV}_\ell(\mathcal{F})$ is shown in case (i) as follows. The claim is obvious if $\text{ADV}_\ell(\mathcal{F}) = \infty$. For all other function classes the claim is shown by induction on $\text{ADV}_\ell(\mathcal{F})$.

We define a learning algorithm B for $\mathcal{F}$, which uses as first hypothesis a function $h : X \to Y$ that assigns to each $x \in X$ some $y \in Y$ such that $\text{ADV}_\ell(\{f \in \mathcal{F} : f(x) = y\})$ is as large as possible. The subsequent Lemma 2 implies that $\text{ADV}_\ell(\{f \in \mathcal{F} : f(x_0) = y_0\}) < \text{ADV}_\ell(\mathcal{F})$ for any pair $\langle x_0, y_0 \rangle$ that the learner might receive as a counterexample to this hypothesis h (because either $\text{ADV}_\ell(\{f \in \mathcal{F} : f(x_0) = y\}) < \text{ADV}_\ell(\mathcal{F})$ for all $y \in Y$, or $h(x_0)$ is the only value $y \in Y$ such that $\text{ADV}_\ell(\{f \in \mathcal{F} : f(x_0) = y\}) = \text{ADV}_\ell(\mathcal{F})$).

For subsequent rounds the learning algorithm B employs a learning algorithm $\tilde{B}$ for $\{f \in \mathcal{F} : f(x_0) = y_0\}$ with $\text{LC}_\ell(\tilde{B}) \leq \text{ADV}_\ell(\{f \in \mathcal{F} : f(x_0) = y_0\})$. Such $\tilde{B}$ exists by the induction hypothesis. Furthermore the preceding facts imply that $\text{LC}_\ell(B) \leq \text{ADV}_\ell(\{f \in \mathcal{F} : f(x_0) = y_0\}) + 1 \leq \text{ADV}_\ell(\mathcal{F})$.

LEMMA 2 *Assume that $A = Y$ and ℓ is any loss function such that $\ell(\tilde{A}, y) > 0$ for any $\tilde{A} \subseteq Y$, $y \in Y$ with $y \notin \tilde{A}$. Then for all $x \in X$ there exists at most one $y \in Y$ such that $\text{ADV}_\ell(\{f \in \mathcal{F} : f(x) = y\}) = \text{ADV}_\ell(\mathcal{F})$.*

Proof of Lemma 2: Assume for a contradiction that for some $x_0 \in X$ there are $y_1, y_2 \in Y$ with $y_1 \neq y_2$ and $\text{ADV}_\ell(\{f \in \mathcal{F} : f(x_0) = y_i\}) = \text{ADV}_\ell(\mathcal{F})$ for $i = 1, 2$.

Then one can build in the following way an adversary tree $\mathcal{U}$ for $\mathcal{F}$ with $\text{INF}_\ell(\mathcal{U}) > \text{ADV}_\ell(\mathcal{F})$: The root of $\mathcal{U}$ is labeled by $\langle x_0, \mathcal{F} \rangle$. Partition Y into A_1, A_2 such that $y_i \notin A_i$ for $i = 1, 2$. Assign $\langle A_1, y_1 \rangle$ and $\langle A_2, y_2 \rangle$ as labels to the edges from the root of $\mathcal{U}$. Attach below the edge with label $\langle A_i, y_i \rangle$ an adversary tree $\mathcal{U}_i$ for $\{f \in \mathcal{F} : f(x_0) = y_i\}$ with $\text{INF}_\ell(\mathcal{U}_i) = \text{ADV}_\ell(\{f \in \mathcal{F} : f(x_0) = y_i\})$. ■

Application 1: Randomization in an oblivious environment

We consider in Theorem 2 an alternative model for function learning, where the learning algorithm may use randomization, and the given sequence of pairs $\langle x, T(x) \rangle$ is fixed before learning takes place and is therefore oblivious to the actions (and the randomization) of the learner (see (Maass, 1991)). For this model we write $\text{RLC-OBL}_\ell(\mathcal{F}, \mathcal{H})$ in place of $\text{LC}_\ell(\mathcal{F}, \mathcal{H})$. It is easy to generalize the results from (Maass, 1991) to show that in the case ℓ is the discrete loss function, $\text{RLC-OBL}_\ell(\mathcal{F}, \mathcal{F}) \leq \ln |\mathcal{F}|$ for any finite function class $\mathcal{F}$. The following limits the amount randomization can help the learner, even in an oblivious environment.

THEOREM 2 *Assume that X, Y, A, $\mathcal{F}$ and ℓ satisfy the assumption of Theorem 1. Then $\text{RLC-OBL}_\ell(\mathcal{F}, A^X) \geq \frac{1}{2}\text{LC-ARB}_\ell(\mathcal{F})$. This lower bound is optimal.*

Proof of Theorem 2: The proof of the lower bound proceeds by showing that any adversary tree $\mathcal{U}$ for $\mathcal{F}$ gives rise to an oblivious sequence S of examples, for which the expected total loss of any randomized learning algorithm is at least $\frac{1}{2}\mathrm{INF}_\ell(\mathcal{U})$.

Let $\mathcal{U}$ be some adversary tree for $\mathcal{F}$ and let B be a probabilistic learning algorithm for $\mathcal{F}$ in the model RLC-OBL with arbitrary hypotheses. One uses $\mathcal{U}$ to construct an oblivious sequence S of adversary responses $\langle x, y\rangle$ such that the expected total loss of B for this sequence S is $\geq \frac{1}{2} \cdot \mathrm{INF}_\ell(\mathcal{U})$. The simultaneous construction of S and of an associated path in $\mathcal{U}$ starts at the root of $\mathcal{U}$. Assume that the so far constructed path ends at an interior node ν with label $\langle x_\nu, \mathcal{F}_\nu\rangle$, and that the two outgoing edges from ν have labels $\langle A_i, y_i\rangle$ for $i = 1, 2$. If we have $p \geq \frac{1}{2}$ for the probability p that $h(x_\nu) \in A_1$ conditioned on the assumption that B has processed the previously constructed sequence of adversary responses (where $h \in A^X$ is the current hypothesis of the learner), then $\langle x_\nu, y_1\rangle$ is chosen to be the next pair in the sequence S. If $p < \frac{1}{2}$ one chooses $\langle x_\nu, y_2\rangle$ as the next pair in S. One extends the definition of the path in $\mathcal{U}$ by the corresponding edge with label $\langle A_i, y_i\rangle$, $i \in \{1, 2\}$. The expected loss of B for this round of the learning process is $\geq \frac{1}{2}\ell(A_1, y_1)$, respectively $\geq \frac{1}{2}\ell(A_2, y_2)$. This implies the claimed lower bound for the expected total loss of B, since the latter is equal to the sum of the expected losses of B for the individual rounds of the learning process.

It was shown in (Maass, 1991) that there exist function classes $\mathcal{F}$ (in fact: classes of $\{0, 1\}$ valued functions) such that $\mathrm{RLC\text{-}OBL}_\ell(\mathcal{F}) = \frac{1}{2}\mathrm{ADV}_\ell(\mathcal{F})$. ■

Application 2: The utility of generalized membership queries

From this point on, we will focus exclusively on the case in which ℓ is the discrete loss function, and we will drop the subscript ℓ from our notation (i.e., writing $\mathrm{LC\text{-}ARB}(\mathcal{F})$ for $\mathrm{LC\text{-}ARB}_\ell(\mathcal{F})$).

We now turn to generalizations of membership queries for function learning. If one allows queries of the form "What is the value of $T(x)$?", and the range Y is infinite, one cannot expect to get a lower bound on the required number of queries in terms of a nontrivial function of $\mathrm{LC\text{-}ARB}_\ell(\mathcal{F})$ that holds for all $\mathcal{F}$. The reason is that the domain X might contain some special point x_0 such that, for any $f \in \mathcal{F}$, $f(x_0)$ reveals the identity of f. The following result provides a lower bound for learning with a large variety of other queries. Special cases of such queries are for example: "In which of the intervals $I_1, \ldots, I_q$ of $\mathbb{R}$ does $T(x_0)$ lie?" (for some partition $I_1, \ldots, I_q$ of the range of f), or "Does T have $1, 2, \ldots, q-1$, or more than $q-1$ points x where the derivative $T'(x)$ of T has value 0"?

The following result shows that the use of arbitrary queries with at most q different possible answers, where $q \leq 1 + \mathrm{LC\text{-}ARB}(\mathcal{F})$, can at best reduce the learning complexity by a factor $\frac{1}{\log(1+\mathrm{LC\text{-}ARB}(\mathcal{F}))}$.

THEOREM 3 *Assume X and Y are arbitrary sets and $\mathcal{F}$ is an arbitrary subset of Y^X (ℓ is the discrete loss function, which is dropped from our notation). We assume that $d := \mathrm{LC\text{-}ARB}(\mathcal{F}) < \infty$.*

Let B be an arbitrary learning algorithm for $\mathcal{F}$ that either proceeds at each round as usual (i.e. B proposes a hypothesis $h \in Y^X$), or asks a query of the form "To which of the subclasses $P_1, \ldots, P_q$ of $\mathcal{F}$ does T belong?", for some arbitrary partition $\langle P_1, \ldots, P_q \rangle$ of $\mathcal{F}$ with $q \leq d+1$. We define the total loss of B in a learning process for $\mathcal{F}$ as the number of queries asked by B plus the sum of the losses of B at the other rounds (i.e. the number of incorrect predictions of B). Let $\mathrm{LC}(B)$ *be the supremum of the total losses of B that occur in learning processes for $\mathcal{F}$.*

Then $\mathrm{LC}(B) \geq \frac{d}{\log(1+d)}$. *This lower bound is optimal.*

Proof of Theorem 3: The proof of the lower bound proceeds by constructing an adversary strategy for B with the help of an adversary tree $\mathcal{U}$ such that $\mathrm{INF}(\mathcal{U}) = \mathrm{LC\text{-}ARB}(\mathcal{F})$.

Fix some adversary tree $\mathcal{U}$ for $\mathcal{F}$ with $\mathrm{INF}(\mathcal{U}) = d$. Select some $f_\nu \in \mathcal{F}_\nu$ for each node ν on level d, and let $\tilde{\mathcal{F}}$ be the set of these 2^d functions f_ν. We define with the help of $\mathcal{U}$ an adversary strategy for the learning algorithm B such that after i rounds there are $\geq 2^d/(d+1)^i$ functions in $\tilde{\mathcal{F}}$ that are consistent with all responses given during the first i rounds (see Theorem 6.8 in [MT 92] for a similar argument).

In the case of a query with a partition $\langle P_1, \ldots, P_q \rangle$ of $\mathcal{F}$ at round $i+1$ the adversary responds with that P_j which contains the largest number of functions in $\tilde{\mathcal{F}}$ that are consistent with all preceding responses.

In the case of an equivalence query at round $i+1$, where the learner proposes a hypothesis $h : X \to Y$, the adversary fixes some path P_h from the root of $\mathcal{U}$ to some node ν_h on level d such that for every node $\nu \neq \nu_h$ on the path P_h the label $\langle \tilde{A}, y \rangle$ of the *other* edge from ν (i.e. that edge which does *not* lie on P_h) satisfies $h(x_\nu) \neq y$. Let $s > 1$ be the number of functions in $\tilde{\mathcal{F}}$ that are consistent with all responses during the first i rounds.

We will subsequently show that for *some* node $\tilde{\nu}$ on path P_h the immediate subtree of $\tilde{\nu}$ that does not contain ν_h contains at least $s/(d+1)$ nodes ν on level d such that $f_\nu \in \tilde{\mathcal{F}}$ is consistent with all responses during the first i rounds. Let $\tilde{\nu}$ be the first such node on P_h, and let $\langle \tilde{A}, y \rangle$ be the label of that edge from $\tilde{\nu}$ that does *not* lie on P_h. Then the adversary gives the response $\langle x_{\tilde{\nu}}, y \rangle$ at round $i+1$. By construction, this response keeps at least $s/(d+1)$ functions in $\tilde{\mathcal{F}}$ "alive" (i.e. consistent with all responses during the first $i+1$ rounds).

Assume for a contradiction that there does not exist any node $\tilde{\nu}$ on P_h as above. We then have $s/(d+1) > 1$, i.e. $s > d+1$, since otherwise f_{ν_h} would be the only function in $\tilde{\mathcal{F}}$ that is consistent with the first i responses (contradicting our assumption that $s > 1$). Since P_h has length d, there are then, apart from f_{ν_h}, less than $d \cdot \frac{s}{d+1}$ functions in $\tilde{\mathcal{F}}$ that are consistent with the first i responses. Hence the definition of s implies that $s < 1 + d \cdot \frac{s}{d+1} < \frac{s}{d+1} + d \cdot \frac{s}{d+1} = s$, a contradiciton. Thus we have shown that the preceding adversary strategy is welldefined.

A straightforward induction on i shows that for every round i there are $\geq 2^d/(d+1)^i$ functions in $\tilde{\mathcal{F}}$ that are consistent with all responses which the preceding adversary strategy has given during the first i rounds.

The optimality of the lower bound of Theorem 3 is demonstrated by the following example. Consider any $d \in \mathbb{N}$ such that $\log_2(d+1) \in \mathbb{N}$. Set $X := \{1, ..., d\}$, $Y := \{0, 1\}$, $\mathcal{F} := Y^X$. Thus LC-ARB$(\mathcal{F}) = d$. In order to construct a learning algorithm B with LC$(B) = \left\lceil \frac{d}{\log_2(d+1)} \right\rceil$, we partition X into $\left\lceil \frac{d}{\log_2(d+1)} \right\rceil$ subsets S_i of size at most $\log_2(d+1)$.

At round i of any learning process, the algorithm asks: "To which of the subclasses $P_1, ..., P_{d+1}$ of $\mathcal{F}$ does T belong?" for a partition $P_1, ..., P_{d+1}$ of $\mathcal{F}$ that classifies each $f \in \mathcal{F}$ according to its restriction to S_i (in other words: at round i the learner asks to see the values of the target function for all arguments in S_i).

■

Remark. The special cases of the Theorems in this section for concepts (i.e. $Y := \{0, 1\}$) are due to Littlestone (1988) (Theorem 1), Littlestone and Maass (Maass, 1991) (Theorem 2), and Maass and Turan (1992) (Theorem 3). Using methods from (Auer and Long, 1994), one gets a suboptimal but more general bound than that in Theorem 3, proving that LC$(B) \geq \frac{d}{\log k} \log\left(\frac{2k}{k+1}\right)$ if queries of the form "To which of the subclasses $P_1, \ldots, P_k$ of $\mathcal{F}$ does T belong?" are allowed for some fixed $k \geq 2$.

Adversary trees for function learning in models with weaker reinforcement

We briefly consider here two natural variations of the model for function learning that was defined at the beginning of this section. It is curious that for each of these two variations there also exists in addition to the usual min-max definition an equivalent max-min definition of the resulting learning complexity in terms of adversary trees.

Let $h_s \in \mathcal{H}$ be the hypothesis of the learner at round s of a learning process. In the first variation of the learning model, we assume that instead of a pair $\langle x_s, T(x_s) \rangle$, the learner receives at round s from the environment just an argument $x_s \in X$ such that $h_s(x_s) \neq T(x_s)$. However the learner does *not* receive the "correct value" $T(x_s)$ for argument x_s. We write LC-ARB$_{\text{weak}}(\mathcal{F})$ for the maximal number of steps that the best learning algorithm for $\mathcal{F}$ with arbitrary hypotheses has to use in this model.

For this learning model we consider a notion of a finite adversary tree $\mathcal{U}_{\text{weak}}$ where each node is labeled by some $x \in X$. Each interior node has fan-out $|Y|$, and its outgoing edges are labeled by all possible values $y \in Y$. We say that a function $f \in \mathcal{F}$ is *qualified* for an edge with label y from a node labeled x if $f(x) \neq y$. We demand that for every leaf of $\mathcal{U}_{\text{weak}}$ there exists at least one function $f \in \mathcal{F}$ that is qualified for all edges on the path from the root to that leaf. We set

$$\mathrm{INF}(\mathcal{U}_{\mathrm{weak}}) := \min\{m \in \mathbb{N} : m \text{ is the length of a path from the root to a leaf in } \mathcal{U}_{\mathrm{weak}}\},$$

and

$$\mathrm{ADV}_{\mathrm{weak}}(\mathcal{F}) := \sup\{\mathrm{INF}(\mathcal{U}_{\mathrm{weak}}) : \mathcal{U}_{\mathrm{weak}} \text{ is an adversary tree for } \mathcal{F} \text{ as defined above}\}.$$

THEOREM 4 *Let X, Y be arbitrary sets (finite or infinite) and $\mathcal{F} \subseteq Y^X$ nonempty. Then*

$$\text{LC-ARB}_{\mathrm{weak}}(\mathcal{F}) = \mathrm{ADV}_{\mathrm{weak}}(\mathcal{F}).$$

Proof of Theorem 4: It is obvious that any adversary tree $\mathcal{U}_{\mathrm{weak}}$ defines an adversary strategy that forces the learner to make at least $\mathrm{INF}(\mathcal{U}_{\mathrm{weak}})$ mistakes. This implies that $\text{LC-ARB}_{\mathrm{weak}}(\mathcal{F}) \geq \mathrm{ADV}_{\mathrm{weak}}(\mathcal{F})$. In order to show the other inequality we set $\mathcal{F}_{x,y} := \{f \in \mathcal{F} : f(x) \neq y\}$. We define a learning algorithm which always uses as next hypothesis h a function such that for any $x \in X$ the value $h(x) \in Y$ has the property that $\mathrm{ADV}_{\mathrm{weak}}(\mathcal{F}_{x,h(x)}) < \mathrm{ADV}_{\mathrm{weak}}(\mathcal{F})$. It will be shown in Lemma 3 that this learning algorithm is well-defined. A straightforward induction on $\mathrm{ADV}_{\ell}(\mathcal{F})$ shows that for every nonempty function class $\mathcal{F} \subseteq Y^X$ this learning algorithm for $\mathcal{F}$ requires at most $\mathrm{ADV}_{\ell}(\mathcal{F})$ learning steps.

LEMMA 3 $\forall x \in X \;\; \exists y \in Y(\mathrm{ADV}_{\mathrm{weak}}(\mathcal{F}_{x,y}) < \mathrm{ADV}_{\mathrm{weak}}(\mathcal{F}))$.

Proof of Lemma 3: Assume that the claim does not hold for some $x_0 \in X$, i.e. $\forall y \in Y(\mathrm{ADV}_{\mathrm{weak}}(\mathcal{F}_{x_0,y}) \geq \mathrm{ADV}_{\mathrm{weak}}(\mathcal{F}))$. Then one can build an adversary tree $\mathcal{U}_{\mathrm{weak}}$ for $\mathcal{F}$ with x_0 as label of the root which satisfies $\mathrm{INF}(\mathcal{U}_{\mathrm{weak}}) > \mathrm{ADV}_{\mathrm{weak}}(\mathcal{F})$. This provides a contradiction to the definition of $\mathrm{ADV}_{\mathrm{weak}}(\mathcal{F})$. ■

The second variation of our function learning model, considered previously by Barland (1992), applies to arbitrary function classes $\mathcal{F} \subseteq Y^X$ for linearly ordered sets Y. In this model the learner receives at round s together with an argument x_s such that $h_s(x_s) \neq T(x_s)$ also the information whether $T(x_s) > h_s(x_s)$ ("$h_s(x_s)$ is too low") or $T(x_s) < h_s(x_s)$ ("$h_s(x_s)$ is too high"). We write $\text{LC-ARB}_{\mathrm{sign}}(\mathcal{F})$ for the maximal number of steps that the best learning algorithm for $\mathcal{F}$ with arbitrary hypotheses has to use in this model.

We define for this model a notion of a finite adversary tree $\mathcal{U}_{\mathrm{sign}}$ where again each node is labeled by some $x \in X$. If a node is not a leaf then it has two outgoing edges that are labeled by "$\geq y$" respectively "$< y$" for some $y \in Y$. We say that a function f is *qualified* for an edge with label "$\geq y$" ("$< y$") from some node with label x if $f(x) \geq y$ (respectively $f(x) < y$). We demand that for every leaf of $\mathcal{U}_{\mathrm{sign}}$

there exists at least one $f \in \mathcal{F}$ that qualifies for all edges on the path from the root to that leaf.

$$\begin{aligned}\mathrm{INF}(\mathcal{U}_{\mathrm{sign}}) := \min\{m \in \mathbb{N} : \ & m \text{ is the length of a path} \\ & \text{from the root to a leaf in } \mathcal{U}_{\mathrm{sign}}\},\end{aligned}$$

and

$$\begin{aligned}\mathrm{ADV}_{\mathrm{sign}}(\mathcal{F}) := \sup\{\mathrm{INF}(\mathcal{U}_{\mathrm{sign}}) : \ & \mathcal{U}_{\mathrm{sign}} \text{ is an adversary tree} \\ & \text{for } \mathcal{F} \text{ as defined above}\}.\end{aligned}$$

THEOREM 5 *Let X be an arbitrary set and let Y be an arbitrary linearly ordered set (finite or infinite). Then for any nonempty function class $\mathcal{F} \subseteq Y^X$,*

$$\text{LC-ARB}_{\mathrm{sign}}(\mathcal{F}) = \mathrm{ADV}_{\mathrm{sign}}(\mathcal{F}).$$

Proof of Theorem 5: Each adversary tree $\mathcal{U}_{\mathrm{sign}}$ defines an adversary strategy in the obvious way. Simultaneously each learning process also defines a path from the root to a leaf in $\mathcal{U}_{\mathrm{sign}}$ in the following way. Assume that one is currently at a node with label x, and that "$\geq y$" and "$< y$" are the labels of the two edges out of that node. If $h(x) < y$, where $h(x)$ is the learner's prediction of the value of the target function at argument x, then the adversary responds that "$h(x)$ is too low " and moves from the node with label x along the edge with label "$\geq y$". Else the adversary responds that "$h(x)$ is too high" and moves along the edge with label "$< y$". This observation implies that $\text{LC-ARB}_{\mathrm{sign}}(\mathcal{F}) \geq \mathrm{ADV}_{\mathrm{sign}}(\mathcal{F})$.

In order to show that $\text{LC-ARB}_{\mathrm{sign}}(\mathcal{F}) \leq \mathrm{ADV}_{\mathrm{sign}}(\mathcal{F})$ we first note that

$$\mathrm{ADV}_{\mathrm{sign}}(\mathcal{F}) = \infty$$

in case that there exists some $x_0 \in X$ with $|\{f(x_0) : f \in \mathcal{F}\}| = \infty$. Hence the claimed inequality is trivial for this case. In the following we consider the case where $|\{f(x) : f \in \mathcal{F}\}|$ is finite for all $x \in X$. The claim is also trivial if $\mathrm{ADV}_{\mathrm{sign}}(\mathcal{F}) = 0$.

We define a learning algorithm for $\mathcal{F}$ that needs at most $\mathrm{ADV}_{\mathrm{sign}}(\mathcal{F})$ steps by recursion over $\mathrm{ADV}_{\mathrm{sign}}(\mathcal{F})$. The algorithm predicts for each $x \in X$ some value $h(x) \in Y$ such that both $\mathrm{ADV}_{\mathrm{sign}}(\{f \in \mathcal{F} : f(x) > h(x)\})$ and $\mathrm{ADV}_{\mathrm{sign}}(\{f \in \mathcal{F} : f(x) < h(x)\})$ are less than $\mathrm{ADV}_{\mathrm{sign}}(\mathcal{F})$. If the adversary responds that "$h(x)$ is too low" ("$h(x)$ is too high") then one generates the next hypothesis in the same manner, but applied to $\{f \in \mathcal{F} : f(x) > h(x)\}$ (respectively $\{f \in \mathcal{F} : f(x) < h(x)\}$) instead of $\mathcal{F}$. One can easily show by induction on $\mathrm{ADV}_{\mathrm{sign}}(\mathcal{F})$ that this learning algorithm uses at most $\mathrm{ADV}_{\mathrm{sign}}(\mathcal{F})$ steps. The following Lemma shows that this algorithm is well-defined.

LEMMA 4 *For all $x \in X$ there exists some $y \in Y$ such that both* $\mathrm{ADV}_{\mathrm{sign}}(\{f \in \mathcal{F} : f(x) > h(x)\})$ *and* $\mathrm{ADV}_{\mathrm{sign}}(\{f \in \mathcal{F} : f(x) < h(x)\})$ *are less then* $\mathrm{ADV}_{\mathrm{sign}}(\mathcal{F})$.

Proof of Lemma 4: Assume that the claim does not hold for some $x_0 \in X$. Since $|\{f(x_0) : f \in \mathcal{F}\}|$ is finite there exists some minimal $y_0 \in Y$ such that $\mathrm{ADV}_{\mathrm{sign}}(\{f \in \mathcal{F} : f(x_0) > y_0\}) < \mathrm{ADV}_{\mathrm{sign}}(\mathcal{F})$. Together with our assumption this implies that $\mathrm{ADV}_{\mathrm{sign}}(\{f \in \mathcal{F} : f(x_0) < y_0\}) \geq \mathrm{ADV}_{\mathrm{sign}}(\mathcal{F})$. Furthermore $\mathrm{ADV}_{\mathrm{sign}}(\{f \in \mathcal{F} : f(x_0) \geq y_0\}) \geq \mathrm{ADV}_{\mathrm{sign}}(\mathcal{F})$ by the minimal choice of y_0. Hence one can build an adversary tree $\mathcal{U}_{\mathrm{sign}}$ for $\mathcal{F}$ with x_0 as label of its root which satisfies $\mathrm{INF}(\mathcal{U}_{\mathrm{sign}}) > \mathrm{ADV}_{\mathrm{sign}}(\mathcal{F})$. This provides a contradiction to the definition of $\mathrm{ADV}_{\mathrm{sign}}(\mathcal{F})$. ■

Remark. It is obvious that for any $\mathcal{F}$,

$$\text{LC-ARB}_{\mathrm{weak}}(\mathcal{F}) \geq \text{LC-ARB}_{\mathrm{sign}}(\mathcal{F}) \geq \text{LC-ARB}(\mathcal{F}).$$

Very recently a new proof technique was introduced in (Auer and Long, 1994) which shows that for any $\mathcal{F} \subseteq Y^X$, $\text{LC-ARB}_{\mathrm{weak}}(\mathcal{F}) \leq 1.39|Y|\lceil 1 + \log_2 |Y|\rceil \text{LC-ARB}(\mathcal{F})$.

3. Learning Unions of Function Classes

In this and the following sections we will only consider the case of function learning with the discrete loss function ℓ. Hence we can delete the subscript ℓ in our notation (as already done in the last part of the preceding section, starting at Application 2).

The quantity $\text{LC-ARB}(\mathcal{F})$ measures the difficulty of learning given that the target function is in $\mathcal{F}$. If one has determined interesting classes $\mathcal{F}_1, ..., \mathcal{F}_p$, then the assumption that the target function is in $\cup_{i=1}^{p} \mathcal{F}_i$ is intuitively safer than assuming that the target is in one of the individual classes. If $\text{LC-ARB}(\cup_{i=1}^{p} \mathcal{F}_i)$ is not much larger than the individual $\text{LC-ARB}(\mathcal{F}_i)$'s, this provides evidence in support of using the optimal algorithm for $\text{LC-ARB}(\cup_{i=1}^{p} \mathcal{F}_i)$.

In this section, we approximately determine the best general bounds on

$$\text{LC-ARB}(\cup_{i=1}^{p} \mathcal{F}_i)$$

obtainable in terms of

$$\text{LC-ARB}(\mathcal{F}_1), \ldots, \text{LC-ARB}(\mathcal{F}_p).$$

To the best of our knowledge, these results are new even for concept learning, where $Y = \{0, 1\}$.

Define φ as follows. For any $k_1, \ldots, k_p$, let

$$\varphi(k_1, ..., k_p) = \max\{\text{LC-ARB}(\cup_{i=1}^{p} \mathcal{F}_i) : \mathcal{F}_1, ..., \mathcal{F}_p \text{ are function classes with } \text{LC-ARB}(\mathcal{F}_i) = k_i \text{ for } i = 1, \ldots, p\}.$$

First, in the case $p = 2$, we can calculate φ exactly.

THEOREM 6 *For all* $k_1, k_2 \geq 0$, $\varphi(k_1, k_2) = k_1 + k_2 + 1$.

Proof: For the upper bound, consider the algorithm which simply uses the optimal algorithm for $\mathcal{F}_1$ until it has received $k_1 + 1$ counterexamples, at which time it switches to the optimal algorithm for $\mathcal{F}_2$.

Now for the lower bound, choose $k_1, k_2 \geq 0$. Let $\mathcal{F}$ be the set of all functions f from $\{1, ..., k_1 + k_2 + 1\}$ to $\{0, 1\}$ such that $|f^{-1}(0)| \leq k_1$. Let $\mathcal{G}$ be the set of all functions g from $\{1, ..., k_1 + k_2 + 1\}$ to $\{0, 1\}$ such that $|g^{-1}(1)| \leq k_2$. Note that every function from $\{1, ..., k_1 + k_2 + 1\}$ to $\{0, 1\}$ must be in either $\mathcal{F}$ or $\mathcal{G}$. In order words, $\mathcal{F} \cup \mathcal{G}$ is the set of all functions from $\{1, ..., k_1 + k_2 + 1\}$ to $\{0, 1\}$ and therefore,

$$\text{LC-ARB}(\mathcal{F} \cup \mathcal{G}) = k_1 + k_2 + 1. \tag{3.1}$$

It is easy to see that the VC-dimensions of $\mathcal{F}$ and $\mathcal{G}$ are k_1 and k_2 respectively, proving that

$$\text{LC-ARB}(\mathcal{F}) \geq k_1 \tag{3.2}$$

$$\text{LC-ARB}(\mathcal{G}) \geq k_2. \tag{3.3}$$

Next, consider the algorithm for $\mathcal{F}$ which repeatedly hypothesizes the function which evaluates to 1 on all previously unseen elements of the domain, therefore only receiving counterexamples which evaluate to 0. Since there are only k_1 such points, LC-ARB$(\mathcal{F}) \leq k_1$. Similarly the algorithm for $\mathcal{G}$ which repeatedly hypothesizes the function which evaluates to 0 on all previously unseen points receives at most k_2 counterexamples. Combining this with (3.2) and (3.3), we can see that LC-ARB$(\mathcal{F}) = k_1$ and LC-ARB$(\mathcal{G}) = k_2$. Combining this with (3.1) yields $\varphi(k_1, k_2) \geq k_1 + k_2 + 1$, completing the proof. ■

The case in which the number p of functions in the union might be greater than 2 is more difficult. We obtain closed-form bounds for this case that match to within a constant factor, and bounds which are sometimes tighter that are not closed-form.

3.1. Closed-form bounds

To establish the closed-form bound, we have found it useful to generalize some of the Weighted Majority (Littlestone and Warmuth, 1991) results to functions with arbitrary ranges.

Suppose we have (LC-ARB) algorithms $A_1, ..., A_n$ for some class $\mathcal{F}$ of functions from X to Y. The *weighted maximum* algorithm (for $A_1, ..., A_n$) studied in this section (we will hereafter refer to it simply as WM) works as follows.

It initializes a sequence $w_{1,1}, ..., w_{1,n}$ of weights. Its initial hypothesis $h_{1,WM}$ is formed as follows. Suppose $h_{1,1}, ..., h_{1,n}$ are the initial hypotheses of $A_1, ..., A_n$ respectively. Then for any $x \in X$, $h_{1,WM}(x)$ is the element $y \in Y$ for which $\sum_{i:h_{1,i}(x)=y} w_{1,i}$ is maximized.

After the tth counterexample (x_t, y_t), the weights are updated as follows:

$$w_{t+1,i} = \begin{cases} w_{t,i} & \text{if } h_{t,i}(x_t) = y_t \\ \frac{w_{t,i}}{2} & \text{otherwise.} \end{cases}$$

Also, for all algorithms A_i for which $h_{t,i}(x_t) \neq y_t$, the counterexample (x_t, y_t) is fed to A_i, and $h_{t+1,i}$ is set to A_i's next hypothesis. For all other algorithms, the hypothesis is unchanged. The master hypothesis $h_{t+1,WM}$ is then calculated from $h_{t+1,1}, ..., h_{t+1,n}$ as above.

The following notation will be useful for the rest of this subsection. For each positive integer t, let

$$v_t = \sum_{i=1}^{n} w_{t,i}$$

be the total weight before the tth counterexample.

Lemma 5 *For any* LC-ARB *algorithms $A_1, ..., A_n$, after m counterexamples,*

$$m \leq \frac{\ln \frac{v_1}{v_{m+1}}}{\ln(4/3)}.$$

Proof: Choose a positive integer $t \leq m$. Let (x_t, y_t) be the tth counterexample received by the WM algorithm. By the definition of $h_{t,WM}$,

$$\sum_{i:h_{t,i}(x_t)=y_t} w_{t,i} \leq \sum_{i:h_{t,i}(x_t)=h_{t,WM}(x_t)} w_{t,i}$$

and therefore, since $y_t \neq h_{t,WM}(x_t)$,

$$\sum_{i:h_{t,i}(x_t)=y_t} w_{t,i} \leq v_t/2. \tag{3.4}$$

Therefore, we have

$$\begin{aligned} v_{t+1} &= \sum_{i=1}^{n} w_{t+1,i} \\ &= \left(\sum_{i:h_{t,i}(x_t)=y_t} w_{t,i} \right) + \left(\sum_{i:h_{t,i}(x_t)\neq y_t} \frac{w_{t,i}}{2} \right) \\ &= \left(\sum_{i:h_{t,i}(x_t)=y_t} w_{t,i} \right) + \frac{1}{2} \left(\sum_{i:h_{t,i}(x_t)\neq y_t} w_{t,i} \right) \\ &= \frac{1}{2} \left(\sum_{i:h_{t,i}(x_t)=y_t} w_{t,i} \right) + \frac{v_t}{2} \\ &\leq \frac{3v_t}{4}, \end{aligned}$$

by (3.4). By induction, we have

$$v_{m+1} \leq (3/4)^m v_1.$$

Solving for m yields the desired result. ■

Many other of the Weighted Majority results (Littlestone and Warmuth, 1991) appear to generalize just as easily.

For each n, define the set SVAR_n of functions from $\{0,1\}^n$ to $\{0,1\}$ to be all $f_i, 1 \leq i \leq n$, where $f_i(\vec{x}) = x_i$ for all $\vec{x} \in \{0,1\}^n$.

Lemma 6 ((Littlestone, 1989)) *For all n,*

$$\text{LC-ARB}(SVAR_n) = \lfloor \log_2 n \rfloor.$$

Now we are ready to establish bounds on φ that match to within a constant factor.

Theorem 7 *For all $p \geq 1, k_1, ..., k_p \geq 0$,*

$$\lfloor \log_2 \sum_{i=1}^{p} 2^{k_i} \rfloor \leq \varphi(k_1, ..., k_p) \leq \frac{1}{\log_2(4/3)} \log_2 \sum_{i=1}^{p} 2^{k_i} \leq 2.41 \log_2 \sum_{i=1}^{p} 2^{k_i}.$$

Proof: We begin with the upper bound. Choose a sequence $\mathcal{F}_1, ..., \mathcal{F}_p$ of classes of functions defined on a common domain X.

Consider the following (LC-ARB) algorithm (call it A) for $\bigcup_{i=1}^{p} \mathcal{F}_i$. It uses the optimal algorithms for $\mathcal{F}_1, ..., \mathcal{F}_p$ (call them $A_1, ..., A_p$) as subroutines for the WM algorithm, with the initial weights set to $2^{\text{LC-ARB}(\mathcal{F}_1)}, ..., 2^{\text{LC-ARB}(\mathcal{F}_p)}$.

Suppose the target is in $\mathcal{F}_i$, and that A has received m counterexamples. Since A_i can receive at most LC-ARB$(\mathcal{F}_i)$ counterexamples, we have

$$v_{m+1} \geq w_{m+1,i} \geq 2^{-\text{LC-ARB}(\mathcal{F}_i)} w_{1,i} = 2^{-\text{LC-ARB}(\mathcal{F}_i)} 2^{\text{LC-ARB}(\mathcal{F}_i)} = 1.$$

Applying Lemma 5,

$$\begin{aligned} m &\leq \frac{\ln\left(\sum_{i=1}^{p} 2^{\text{LC-ARB}(\mathcal{F}_i)}\right)}{\ln(4/3)} \\ &= \frac{\log_2\left(\sum_{i=1}^{p} 2^{\text{LC-ARB}(\mathcal{F}_i)}\right)}{\log_2(4/3)} \\ &\leq 2.41 \log_2 \left(\sum_{i=1}^{p} 2^{\text{LC-ARB}(\mathcal{F}_i)} \right). \end{aligned}$$

Since $\mathcal{F}_1, ..., \mathcal{F}_p$ were chosen arbitrarily,

$$\varphi(k_1, ..., k_p) \leq \frac{1}{\log_2(4/3)} \log_2 \sum_{i=1}^{p} 2^{k_i}. \tag{3.5}$$

Now for the lower bound. Choose $k_1, ..., k_p$. Let $X = \{0,1\}^{\sum_{i=1}^{p} 2^{k_i}}, Y = \{0,1\}$. For each $u \leq \sum_{i=1}^{p} 2^{k_i}$, let $f_u : X \to Y$ be defined by setting $f_u(\vec{x}) = x_u$ for all $\vec{x} \in X$. For each $i \leq p$, let

$$G_i = \left\{ f_u : \sum_{j=1}^{i-1} 2^{k_j} < u \leq \sum_{j=1}^{i} 2^{k_j} \right\}.$$

By trivial application of Lemma 6, for all i, LC-ARB$(G_i) = k_i$, and

$$\text{LC-ARB}(\cup_{i=1}^{p} G_i) = \left\lfloor \log_2 \sum_{i=1}^{p} 2^{k_i} \right\rfloor.$$

Thus, $\varphi(k_1, ..., k_p) \geq \lfloor \log_2 \sum_{i=1}^{p} 2^{k_i} \rfloor$. Combining this with (3.5) completes the proof. ■

Working more on the bounds of Theorem 7, we obtain even looser, but simpler, bounds, which still match to within a constant factor, and give a better idea of the flavor of the behavior of φ.

THEOREM 8 *For all* $p \geq 2, k_1, ..., k_p \geq 0$,

$$\begin{aligned} &\tfrac{1}{2}(\log_2 p + \max_i k_i - 1) \\ &\quad < \varphi(k_1, ..., k_p) \\ &\qquad \leq \tfrac{1}{\log_2(4/3)}(\log_2 p + \max_i k_i) \\ &\qquad \leq 2.41(\log_2 p + \max_i k_i). \end{aligned}$$

Proof: Choose $p \geq 1, k_1, ..., k_p \geq 0$. Then, by Theorem 7,

$$\begin{aligned} \varphi(k_1, ..., k_p) &\leq \frac{1}{\log_2(4/3)} \log_2 \sum_{i=1}^{p} 2^{k_i} \\ &\leq \frac{1}{\log_2(4/3)} \log_2 (p \max_i 2^{k_i}) \\ &= \frac{1}{\log_2(4/3)} (\log_2 p + \max_i k_i) \\ &\leq 2.41(\log_2 p + \max_i k_i) \end{aligned}$$

establishing the upper bound.

Also, again by Theorem 7,

$$\begin{aligned}
\varphi(k_1, ..., k_p) &\geq \lfloor \log_2 \sum_{i=1}^{p} 2^{k_i} \rfloor \\
&> \left(\log_2 \sum_{i=1}^{p} 2^{k_i} \right) - 1 \\
&\geq \left(\log_2 \left((p-1) + \max_i 2^{k_i} \right) \right) - 1 \\
&\geq \log_2 \left(2\sqrt{(p-1)\left(\max_i 2^{k_i} \right)} \right) - 1 \\
&= \frac{1}{2} \left(\log_2 (p-1) + \max_i k_i \right) \\
&\geq \frac{1}{2} \left(\log_2 p + \max_i k_i - 1 \right)
\end{aligned}$$

completing the proof. ■

We next turn to proving that the upper bounds of Theorems 7 and 8 cannot be improved by reducing the leading constant.

For each $u \in \mathbb{N}$, let POWER_u be the set of all functions from $\{1, ..., u\}$ to $\{0, 1\}$ and for each $v \leq u$ and for each $f \in \text{POWER}_u$, let

$$\mathcal{B}_{f,v} = \{ g \in \text{POWER}_u : | \{ x : g(x) \neq f(x) \} | \leq v \}.$$

The following lemma, implicit in the work of Maass and Turán (Maass and Turán, 1992, Proposition 6.3) will prove useful.

Lemma 7 ((Maass and Turán, 1992)) *Define* $p : \mathbb{N} \to \mathbb{N}$ *by*

$$p(k) = \left\lceil 9k^2 2^{(5 - 3\log_2 3)k} \right\rceil.$$

Then for each $k \in \mathbb{N}$*, there exists* $f_1, ..., f_{p(k)} \in POWER_{3k}$ *such that*

$$POWER_{3k} = \bigcup_{i=1}^{p(k)} \mathcal{B}_{f_i,k}.$$

We apply this to show that the $1/\log_2(4/3)$ of Theorems 7 and 8 is optimal.

Theorem 9 *If* $p : \mathbb{N} \to \mathbb{N}$ *is defined as in Lemma 7, then*

$$\varphi(\overbrace{k, ..., k}^{p(k)\ times}) \geq \left(\frac{1}{\log_2(4/3)} - o(1)\right)(\log_2 p(k) + k).$$

Proof: As observed in (Maass and Turán, 1992), for any f, LC-ARB($\mathcal{B}_{f,k}$) $= k$ (predict with f on unseen points). Since LC-ARB(POWER$_{3k}$) $= 3k$,

$$\varphi(\overbrace{k, ..., k}^{p(k) \text{ times}}) \geq 3k. \tag{3.6}$$

Also,

$$\log_2 p(k) + k \leq ((5 - 3\log_2 3)k + o(k)) + k = \left(3\log_2 \frac{4}{3}\right) k + o(k).$$

Hence

$$3k \geq \frac{1}{\log_2 \frac{4}{3}}(\log_2 p(k) + k - o(k)).$$

Combining this with (3.6) yields the desired result. ■

3.2. Other bounds

Now we described some bounds which are sometimes tighter, yet which are not closed form. The following theorem describes the bounds, where $\binom{u}{\leq v}$ is defined to be $\sum_{i=0}^{v} \binom{u}{i}$.

THEOREM 10 *For all* $p \geq 1$, $k_1, ..., k_p \geq 0$,

$$\begin{aligned} &\max \left\{ l \in \mathbb{N}_0 : 2^l \ln 2^l \leq \textstyle\sum_{i=1}^{p} \binom{l}{\leq k_i} \right\} \\ &\quad < \varphi(k_1, ..., k_p) \\ &\quad \leq \max \left\{ l \in \mathbb{N}_0 : 2^l \leq \textstyle\sum_{i=1}^{p} \binom{l}{\leq k_i} \right\}. \end{aligned}$$

These bounds are sometimes tighter than those of Theorem 7. For example, Theorem 7 implies that $\varphi(5, 10, 20) \leq 48$, where Theorem 10 implies that $\varphi(5, 10, 20) \leq 31$.

Our proof proceeds through a series of lemmas.

For the upper bound, we use a technique that was developed for coding theory and extended to address the problem of searching using a comparator that occasionally lies ((Berlekamp, 1968), (Rivest, Meyer, Kleitman, Winklman and Spencer, 1980), (Spencer, 1992)). Cesa-Bianchi, Freund, Helmbold and Warmuth (1994) showed that the proof of (Spencer, 1992) could be modified to determine the (in some cases only nearly) best bound on the number of mistakes for learning a class of $\{0, 1\}$-valued functions, given that an unknown algorithm from a pool $\{A_1, ..., A_p\}$ would make at most k mistakes. Due to the fact that the functions in our application take on potentially more than two values, the elegant argument from (Cesa-Bianchi, Freund, Helmbold and Warmuth, 1994) cannot easily be modified for our application. An inductive argument like that given below is apparently required.

The lower bound is obtained by generalizing Lemma 7 and the argument of Theorem 9.

For the remainder of this section, we adopt the convention that LC-ARB$(\emptyset) = -1$, and that for all integers $u \geq 0, w < 0$, $\binom{u}{w} = 0$. Note that even with the above convention, the following standard relationships still hold.

LEMMA 8 *For all integers u, v such that $u \geq 1$,*

$$\binom{u}{v} = \binom{u-1}{v} + \binom{u-1}{v-1}$$

and

$$\binom{u}{\leq v} = \binom{u-1}{\leq v} + \binom{u-1}{\leq v-1}.$$

We begin by establishing the upper bound. We will make use of a general-purpose algorithm described in the following definition.

Definition. For each sequence $\mathcal{F}_1, ..., \mathcal{F}_p$ of function classes, define the algorithm $B_{\mathcal{F}_1,...,\mathcal{F}_p}$ for learning $\cup_{i=1}^p \mathcal{F}_i$ as follows. Suppose for each i, $k_i =$ LC-ARB$(\mathcal{F}_i)$. Let

$$r = \max \left\{ l \in \mathbb{N}_0 : 2^l \leq \sum_{i=1}^p \binom{l}{\leq k_i} \right\}.$$

(One can interpret r as the mistake bound that $B_{\mathcal{F}_1,...,\mathcal{F}_p}$ is shooting for.) Let $B_1, ..., B_p$ be optimal algorithms for $\mathcal{F}_1, ..., \mathcal{F}_p$ respectively. (Note that in this model, optimal algorithms exist.) Then $B_{\mathcal{F}_1,...,\mathcal{F}_p}$'s initial hypothesis h is constructed as follows. If $h_1, ..., h_p$ are the initial hypotheses of $B_1, ..., B_p$ respectively, for each x, choose $h(x)$ arbitrarily so that

$$h(x) \in \left\{ u \in Y : \forall y \sum_{i:h_i(x)=u} \binom{r}{k_i} \geq \sum_{i:h_i(x)=y} \binom{r}{k_i} \right\}.$$

Notice that h can be viewed as taking a "weighted maximum" of the hypotheses $h_1, ..., h_p$, with weights $\binom{r}{k_1}, ..., \binom{r}{k_p}$. After getting a first counterexample (x, y), if for each $i \leq p$, $\mathcal{G}_i = \{f \in \mathcal{F}_i : f(x) = y\}$, $B_{\mathcal{F}_1,...,\mathcal{F}_p}$ recurses to $B_{\mathcal{G}_1,...,\mathcal{G}_p}$ learning $\cup_{i=1}^p \mathcal{G}_i$.

The following lemma shows that $B_{\mathcal{F}_1,...,\mathcal{F}_p}$'s hypothesis minimizes a technical quantity used later in our proofs.

LEMMA 9 *Choose function classes $\mathcal{F}_1, ..., \mathcal{F}_p$. Then if $r, k_1, ..., k_p, h_1, ..., h_p$ and h are defined as in Definition 3.2, for all $x \in X$, if $\hat{y} = h(x)$, then for all $y \in Y$,*

$$\textstyle\sum_{i:h_i(x)=\hat{y}} \binom{r}{\le k_i - 1} + \sum_{i:h_i(x)\ne\hat{y}} \binom{r}{\le k_i}$$
$$\le \textstyle\sum_{i:h_i(x)=y} \binom{r}{\le k_i - 1} + \sum_{i:h_i(x)\ne y} \binom{r}{\le k_i}.$$

Proof: Choose x. Choose y. Then

$$\sum_{i:h_i(x)=y} \binom{r}{k_i} \le \sum_{i:h_i(x)=\hat{y}} \binom{r}{k_i}$$

which implies

$$\sum_{i:h_i(x)=y} \left(\binom{r}{\le k_i} - \binom{r}{\le k_i - 1}\right) \le \sum_{i:h_i(x)=\hat{y}} \left(\binom{r}{\le k_i} - \binom{r}{\le k_i - 1}\right).$$

Rearranging terms yields

$$\textstyle\sum_{i:h_i(x)=\hat{y}} \binom{r}{\le k_i - 1} + \sum_{i:h_i(x)=y} \binom{r}{\le k_i}$$
$$\le \textstyle\sum_{i:h_i(x)=y} \binom{r}{\le k_i - 1} + \sum_{i:h_i(x)=\hat{y}} \binom{r}{\le k_i}.$$

Finally, adding $\sum_{i:h_i(x)\notin\{y,\hat{y}\}} \binom{r}{\le k_i}$ to the second summation on either side of the inequality yields

$$\textstyle\sum_{i:h_i(x)=\hat{y}} \binom{r}{\le k_i - 1} + \sum_{i:h_i(x)\ne\hat{y}} \binom{r}{\le k_i}$$
$$\le \textstyle\sum_{i:h_i(x)=y} \binom{r}{\le k_i - 1} + \sum_{i:h_i(x)\ne y} \binom{r}{\le k_i}$$

completing the proof. ■

The following lemma is well known and easily verified.

LEMMA 10 *Choose integers $l, p \ge 0, k_1, ..., k_p$. Then*

$$\sum_{i=1}^{p} \binom{l+1}{\le k_i} \le 2\sum_{i=1}^{p} \binom{l}{\le k_i}.$$

We establish the upper bound for Theorem 10 in the following lemma.

LEMMA 11 *Choose classes $\mathcal{F}_1, ..., \mathcal{F}_p$ of functions with $\cup_{i=1}\mathcal{F}_i \ne \emptyset$. Then if for $i \le p$, $k_i = \text{LC-ARB}(\mathcal{F}_i)$,*

$$\text{LC-ARB}(\cup_{i=1}^{p}\mathcal{F}_i) \le \max\left\{ l \in \mathbb{N}_0 : 2^l \le \sum_{i=1}^{p} \binom{l}{\le k_i} \right\}.$$

Proof: The proof is by induction on $p + \sum_{i=1}^{p} k_i$. The induction hypothesis is that for each positive integer m, for any $\mathcal{F}_1, ..., \mathcal{F}_p$, if $k_1, ..., k_p$ and r are defined as in Definition 3.2 and satisfy $m = p + \sum_{i=1}^{p} k_i$, then $B_{\mathcal{F}_1,\ldots,\mathcal{F}_p}$ makes at most r mistakes.

Base case: If $p + \sum_{i=1}^{p} k_i = 1$, then $k_j \geq 0$ and $\mathcal{F}_j \neq \emptyset$ for exactly one j, and $k_j = 0$ and $|\mathcal{F}_j| = 1$. Define $r, h, h_1, ..., h_p$ as in Definition 3.2. In this case $r = 0$, $\binom{r}{k_j} = 1$ and $\binom{r}{k_i} = 0$ for all $i \neq j$. Thus $h(x) = h_j(x)$ for all x, and $B_{\mathcal{F}_1,\ldots,\mathcal{F}_p}$ never makes a mistake, establishing the base case.

Induction step: Assume $m > 1$. Choose $\mathcal{F}_1, ..., \mathcal{F}_p$ such that if $k_1, ..., k_p$ are defined as in Definition 3.2, then $m = p + \sum_{i=1}^{p} k_i$. Define $r, h, h_1, ..., h_p$ as in Definition 3.2. Note that $r \geq 1$.

Choose a target function T in $\cup_{i=1}^{p} \mathcal{F}_i$. Choose $x \in X$. Let $y = T(x)$ and $\hat{y} = h(x)$. Assume without loss of generality that $\hat{y} \neq y$. For each $i \leq p$, let $\mathcal{G}_i = \{f \in \mathcal{F}_i : f(x) = y\}$, and $l_i = \text{LC-ARB}(\mathcal{G}_i)$.

$$\begin{aligned}
\sum_{i=1}^{p} \binom{r+1}{\leq k_i} &= \sum_{i=1}^{p} \left(\binom{r}{\leq k_i} + \binom{r}{\leq k_i - 1} \right) \qquad \text{(Lemma 8)} \\
&= \left(\sum_{i:h_i(x)=y} \binom{r}{\leq k_i} + \sum_{i:h_i(x)\neq y} \binom{r}{\leq k_i - 1} \right) \\
&\qquad + \left(\sum_{i:h_i(x)\neq y} \binom{r}{\leq k_i} + \sum_{i:h_i(x)=y} \binom{r}{\leq k_i - 1} \right) \\
&\geq \left(\sum_{i:h_i(x)=y} \binom{r}{\leq k_i} + \sum_{i:h_i(x)\neq y} \binom{r}{\leq k_i - 1} \right) \\
&\qquad + \left(\sum_{i:h_i(x)\neq \hat{y}} \binom{r}{\leq k_i} + \sum_{i:h_i(x)=\hat{y}} \binom{r}{\leq k_i - 1} \right)
\end{aligned}$$

by Lemma 9. Breaking apart one of the summations, we get

$$\begin{aligned}
&\textstyle\sum_{i=1}^{p} \binom{r+1}{\leq k_i} \\
&\quad \geq \left(\textstyle\sum_{i:h_i(x)=y} \binom{r}{\leq k_i} + \sum_{i:h_i(x)\neq y} \binom{r}{\leq k_i-1} \right) \\
&\qquad\quad + \left(\textstyle\sum_{i:h_i(x)=y} \binom{r}{\leq k_i} + \sum_{i:h_i(x)\notin\{\hat{y},y\}} \binom{r}{\leq k_i} + \sum_{i:h_i(x)=\hat{y}} \binom{r}{\leq k_i-1} \right).
\end{aligned} \tag{3.7}$$

Since, for all integers $u \geq 0, v$, $\binom{u}{\leq v} \geq \binom{u}{\leq v-1}$ (recall that $\binom{u}{\leq v}$ is defined to be 0 for negative v), (3.7) implies

$$\begin{aligned}
&\textstyle\sum_{i=1}^{p} \binom{r+1}{\leq k_i} \\
&\quad \geq \left(\textstyle\sum_{i:h_i(x)=y} \binom{r}{\leq k_i} + \sum_{i:h_i(x)\neq y} \binom{r}{\leq k_i - 1}\right) \\
&\qquad + \left(\textstyle\sum_{i:h_i(x)=y} \binom{r}{\leq k_i} + \sum_{i:h_i(x)\notin\{\hat{y},y\}} \binom{r}{\leq k_i - 1} + \sum_{i:h_i(x)=\hat{y}} \binom{r}{\leq k_i - 1}\right) \\
&\quad = 2\left(\textstyle\sum_{i:h_i(x)=y} \binom{r}{\leq k_i} + \sum_{i:h_i(x)\neq y} \binom{r}{\leq k_i - 1}\right).
\end{aligned}$$

Since $B_1, ..., B_p$ are optimal, $l_i \leq k_i$ if $h_i(x) = y$ and $l_i \leq k_i - 1$ if $h_i(x) \neq y$, so

$$\sum_{i=1}^{p} \binom{r+1}{\leq k_i} \geq 2 \sum_{i=1}^{p} \binom{r}{\leq l_i}. \tag{3.8}$$

Since

$$r = \max\left\{l \in \mathbb{N}_0 : 2^l \leq \sum_{i=1}^{p} \binom{l}{\leq k_i}\right\}$$

we have

$$2^{r+1} > \sum_{i=1}^{p} \binom{r+1}{\leq k_i}.$$

Applying (3.8) yields

$$2^{r+1} > 2 \sum_{i=1}^{p} \binom{r}{\leq l_i}$$

which trivially implies

$$2^{r} > \sum_{i=1}^{p} \binom{r}{\leq l_i}.$$

Thus

$$r \notin \left\{l \in \mathbb{N}_0 : 2^l \leq \sum_{i=1}^{p} \binom{l}{\leq l_i}\right\}$$

which, together with Lemma 10, implies

$$\max\left\{l \in \mathbb{N}_0 : 2^l \leq \sum_{i=1}^{p} \binom{l}{\leq l_i}\right\} < r,$$

which in turn gives

$$\max\left\{l \in \mathbb{N}_0 : 2^l \leq \sum_{i=1}^{p} \binom{l}{\leq l_i}\right\} \leq r - 1. \tag{3.9}$$

However, by the induction hypothesis, the number of mistakes made by $B_{\mathcal{G}_1,\ldots,\mathcal{G}_p}$ is at most

$$\max\left\{l \in \mathbb{N}_0 : 2^l \leq \sum_{i=1}^{p} \binom{l}{\leq l_i}\right\},$$

and therefore the number of mistakes made by $B_{\mathcal{F}_1,\ldots,\mathcal{F}_p}$ is at most

$$1 + \max\left\{l \in \mathbb{N}_0 : 2^l \leq \sum_{i=1}^{p} \binom{l}{\leq l_i}\right\}.$$

Combining this with (3.9) completes the proof. ■

The following more general variant of Lemma 7 will be useful in proving lower bounds on φ.

Lemma 12 ((Maass and Turán, 1992)) *Choose integers $k_1, \ldots, k_p \geq 0$. Then for each integer l such that*

$$2^l \ln 2^l < \sum_{i=1}^{p} \binom{l}{\leq k_i} \tag{3.10}$$

there exist $f_1, \ldots, f_p \in POWER_l$ such that

$$POWER_l = \bigcup_{i=1}^{p} \mathcal{B}_{f_i,k_i}. \tag{3.11}$$

Proof: Choose l satisfying (3.10). Let U be the uniform distribution on POWER_l. Choose f. Then for each i,

$$\mathbf{Pr}_{g \in U}(|\{x : f(x) \neq g(x)\}| > k_i) = \left(1 - 2^{-l}\binom{l}{\leq k_i}\right).$$

Thus, for our fixed f,

$$\mathbf{Pr}_{(g_1,\ldots,g_p) \in U^p}(\forall i \ \ |\{x : f(x) \neq g_i(x)\}| > k_i) = \prod_{i=1}^{p}\left(1 - 2^{-l}\binom{l}{\leq k_i}\right).$$

This implies that

$$\begin{aligned} &\mathbf{Pr}_{(g_1,\ldots,g_p) \in U^p}(\exists f \in \mathrm{POWER}_l \ \forall i \ |\{x : f(x) \neq g_i(x)\}| > k_i) \\ &\qquad \leq 2^l \textstyle\prod_{i=1}^{p}\left(1 - 2^{-l}\binom{l}{\leq k_i}\right). \end{aligned} \tag{3.12}$$

We have

$$2^l \ln 2^l < \sum_{i=1}^{p} \binom{l}{\leq k_i}.$$

Dividing by 2^l and exponentiating yields

$$2^l < \exp\left(2^{-l} \sum_{i=1}^{p} \binom{l}{\leq k_i}\right).$$

Dividing both sides by the right hand side and "pushing the exponential function through the sum," we get

$$2^l \prod_{i=1}^{p} \exp\left(-2^{-l} \binom{l}{\leq k_i}\right) < 1.$$

Applying the well known fact that for all x, $1 + x \leq e^x$ yields

$$2^l \prod_{i=1}^{p} \left(1 - 2^{-l} \binom{l}{\leq k_i}\right) < 1.$$

By (3.12), this implies that

$$\mathbf{Pr}_{(g_1,\ldots,g_p)\in U^p}(\exists f \in \mathrm{POWER}_l \forall i |\{x : f(x) \neq g_i(x)\}| > k_i) < 1,$$

and therefore there exist $g_1, ..., g_p$ such that for all $f \in \mathrm{POWER}_l$, there is an i such that $|\{x : f(x) \neq g_i(x)\}| \leq k_i$, i.e. (3.11) holds. This completes the proof.

■

We finish the proof of Theorem 10 by establishing the lower bound.

LEMMA 13 *For all $p \geq 1$, $k_1, ..., k_p \geq 0$,*

$$\varphi(k_1, ..., k_p) > \max\left\{l \in \mathbb{N}_0 : 2^l \ln 2^l \leq \sum_{i=1}^{p} \binom{l}{\leq k_i}\right\}.$$

Proof: Fix $p \geq 1, k_1, ..., k_p \geq 0$. Choose l such that

$$2^l \ln 2^l < \sum_{i=1}^{p} \binom{l}{\leq k_i} \tag{3.13}$$

and let $f_1, ..., f_p \in \mathrm{POWER}_l$ be such that $\mathrm{POWER}_l = \bigcup_{i=1}^{p} \mathcal{B}_{f_i,k_i}$. The existence of such f_i's is guaranteed by Lemma 12. Since for each i, LC-ARB($\mathcal{B}_{f_i,k_i}$) $= k_i$ and LC-ARB(POWER_l) $= l$, we have that

$$\varphi(k_1, ..., k_p) \geq l,$$

and since l was chosen subject to (3.13), we have

$$\varphi(k_1, ..., k_p) \geq \max\left\{ l \in \mathbb{N}_0 : 2^l \ln 2^l < \sum_{i=1}^{p} \binom{l}{\leq k_i} \right\},$$

completing the proof. ■

Suppose φ_{weak} is defined analogously to φ for the LC-ARB$_{\text{weak}}$ model described at the end of Section 2. Formally,

$$\varphi_{\text{weak}}(k_1, ..., k_p) = \max\{\text{LC-ARB}_{\text{weak}}(\cup_{i=1}^{p} \mathcal{F}_i) : \mathcal{F}_1, ..., \mathcal{F}_p \text{ are function classes with} \text{LC-ARB}_{\text{weak}}(\mathcal{F}_i) = k_i \text{ for } i = 1, \ldots, p\}.$$

Using more a straightforward argument, we may determine φ_{weak} exactly.

THEOREM 11 *For all $p \geq 1$, $k_1, ..., k_p \geq 0$,*

$$\varphi_{\text{weak}}(k_1, ..., k_p) = (p-1) + \sum_{i=1}^{p} k_i.$$

Proof: Choose $p \geq 1$, $k_1, ..., k_p \geq 0$.

We begin with the upper bound. Choose a set X and sets $\mathcal{F}_1, ..., \mathcal{F}_p$ for which

$$\text{LC-ARB}_{\text{weak}}(\mathcal{F}_1) = k_1, ..., \text{LC-ARB}_{\text{weak}}(\mathcal{F}_p) = k_p.$$

Consider the following algorithm for learning $\cup_{i=1}^{p} \mathcal{F}_i$. If $p = 1$, it just runs the optimal algorithm for $\mathcal{F}_1$. If $p > 1$, it first runs the optimal algorithm for $\mathcal{F}_1$ until either it correctly guesses the target or until it gets LC-ARB$_{\text{weak}}(\mathcal{F}_1) + 1$ counterexamples. In the second case, it recurses to learning $\cup_{i=2}^{p} \mathcal{F}_i$. By a trivial induction, this algorithm receives at most

$$p - 1 + \sum_{i=1}^{p} \text{LC-ARB}_{\text{weak}}(\mathcal{F}_i)$$

counterexamples, establishing the fact that

$$\varphi_{\text{weak}}(k_1, ..., k_p) \leq (p-1) + \sum_{i=1}^{p} k_i.$$

Now for the lower bound. For each $i \leq p$, define $\mathcal{F}_i$ to be the set of all functions f from $\{1, ..., p - 1 + \sum_{i=1}^{p} k_i\}$ to $\{2i-1, 2i\}$ for which $|f^{-1}(2i)| \leq k_i$. Trivially, for each such i,

$$\text{LC-ARB}_{\text{weak}}(\mathcal{F}_i) = k_i. \tag{3.14}$$

Choose an algorithm A for learning $\cup_{i=1}^{p} F_i$ in the LC-ARB$_{\mathrm{weak}}$ model. Let h_1 be A's initial hypothesis, and for each $1 < t \leq p-1+\sum_{i=1}^{p} k_i$, let h_t be A's hypothesis, given that each previous hypothesis h_s had received s as a counterexample. Define $g : \{1, ..., p-1+\sum_{i=1}^{p} k_i\} \rightarrow \{1, ..., 2p\}$ by $g(t) = h_t(t)$. Let $i_0 \leq p$ be such that

$$|g^{-1}(\{2i_0 - 1, 2i_0\})| \leq k_{i_0}.$$

Note that such an i_0 must exist, since otherwise

$$\begin{aligned} |\{1, ..., p-1+\sum_{i=1}^{p} k_i\}| &= |\cup_{i=1}^{p} g^{-1}(\{2i-1, 2i\})| \\ &= \sum_{i=1}^{p} |g^{-1}(\{2i-1, 2i\})| \\ &\geq \sum_{i=1}^{p} (k_i + 1) \\ &= p + \sum_{i=1}^{p} k_i, \end{aligned}$$

a contradiction. Define $f : \mathbb{N} \rightarrow \{1, ..., 2p\}$ as follows

$$f(x) = \begin{cases} 2i_0 & \text{if } g(x) = 2i_0 - 1 \\ 2i_0 - 1 & \text{otherwise.} \end{cases}$$

Since

$$|g^{-1}(\{2i_0 - 1\})| \leq |g^{-1}(\{2i_0 - 1, 2i_0\})| \leq k_{i_0},$$

we have $f \in F_{i_0}$, and therefore $f \in \cup_{i=1}^{p} F_i$. Furthermore, for each $t \leq p-1+\sum_{i=1}^{p} k_i$

$$h_t(t) = g(t) \neq f(t),$$

and if f is the target, t is a valid counterexample for h_t. Since A was chosen arbitrarily,

$$\text{LC-ARB}_{\mathrm{weak}}(\cup_{i=1}^{p} F_i) \geq p - 1 + \sum_{i=1}^{p} k_i.$$

Combining this with (3.14) yields the desired result. ■

Notice that the range of $\cup_{i=1}^{p} F_i$ used in the lower bound argument above was increasingly large as p got bigger. Thus, for example, determining the best bounds on LC-ARB$_{\mathrm{weak}}(\cup_{i=1}^{p} F_i)$ that can be obtained in terms of

$$\text{LC-ARB}_{\mathrm{weak}}(F_1), ..., \text{LC-ARB}_{\mathrm{weak}}(F_p)$$

and the size of the range of $\cup_{i=1}^{p} F_i$ remains an open problem.

4. Learning the Maximum of Linear Functions

A natural and commonly studied function class is the set of piecewise linear functions. Piecewise linear functions are simple enough to allow efficient algorithms (e.g. linear programming); they are easy to handle and easy to describe. On the other hand, they are sufficiently complex to approximate and model most functions occuring in practice. Unfortunately, the following example shows that no finite mistake bound can be obtained even for the very restricted case of continuous functions consisting of three linear pieces and with arbitrary functions as hypotheses.[2]

EXAMPLE 1 *For* $0 < a < b < 1$*, we define continuous piecewise linear functions* $g_{a,b} : \mathbb{R} \to \mathbb{R}$ *with*

$$g_{a,b}(x) = \begin{cases} 0 & \text{if } x \leq a \\ (x-a)/(b-a) & \text{if } a < x < b \\ 1 & \text{if } b \leq x. \end{cases}$$

An infinite sequence of CEs can be constructed as follows: Let $a_1 = 0$*,* $b_1 = 1$*. For all* $i \geq 1$ *the i-th CE is given by* $((a_i + b_i)/2, f_i)$ *where* $f_i \in \{0, 1\}$ *depending on the learner's hypothesis. If* $f_i = 0$ *then* $a_{i+1} = (a_i + b_i)/2$*,* $b_{i+1} = b_i$*; if* $f_i = 1$ *then* $a_{i+1} = a_i$*,* $b_{i+1} = (a_i + b_i)/2$*. Clearly this process does not terminate and for any i the function* g_{a_i,b_i} *is consistent with all previous CEs.*

The function in Example 1 is non-convex. In this section we will show that under the restriction of *convexity*, the class of piecewise linear functions admits mistake-bounded algorithms, thus yielding one of the first positive results for functions of more than one real variable. We will present two algorithms for this learning problem. The mistake bound for the first algorithm grows polynomially in the number of pieces for a fixed number of variables, and the mistake bound for the second algorithm grows polynomially in the number of variables for a fixed number of pieces. These results trivially imply mistake-bound algorithms for the class of arbitrary continuous functions consisting of two linear pieces, since all such functions are either convex or concave.

For a class $\mathcal{F}$ of functions, we denote by $\mathcal{F}_k^{\max}$ the class $\{f : \exists f_1, ..., f_k \in \mathcal{F} \ \forall x \ (f(x) = \max_{1 \leq i \leq k} f_i(x))\}$. Let $\mathcal{L}(d)$ consist of the linear functions $\mathbb{R}^d \to \mathbb{R}$. (A *linear function* is defined as usual by a vector $c \in \mathbb{R}^d$, $c \neq 0$ and a $b \in \mathbb{R}$. Points $x \in \mathbb{R}^d$ are mapped to $c'x + b$.)

4.1. The maximum of linear functions of a fixed number of variables

We first present the algorithm whose mistake bound is polynomial in the number k of pieces if the number d of variables is constant. Hence, let $T \in \mathcal{L}(d)_k^{\max}$ be the unknown target function, with $T(x) = \max_{1 \leq i \leq k} f_i(x)$ for certain $f_i \in \mathcal{L}(d)$. We begin by defining a technical quantity central in our analysis.

Definition. For integers $k \geq 2$, $\delta \geq 1$, we define $\psi_k(\delta) = (3k^{\delta} - 2k^{\delta-1} - 1)/(k-1)$. (Note that $\psi_k(\delta)$ is integer for all k, δ). Moreover, we define $\psi_k(0) = 1$ and $\psi_1(\delta) = \delta + 1$.

Definition. For $P \subseteq \mathbb{R}^{d+1}$, the *dimension* $\delta(P)$ of P is defined to be the dimension of the smallest dimensional *affine subspace* $S(P)$ of $\mathbb{R}^{d+1}$ with $P \subseteq S(P)$. A set $P \subseteq \mathbb{R}^{d+1}$ is called *safe* with respect to $\mathcal{L}(d)_k^{\max}$, if $|P| \geq \psi_k(\delta(P))$.

Since the intersection of two affine subspaces yields another affine subspace, $\delta(P)$ is well-defined. The reason for dealing with point sets in $\mathbb{R}^{d+1}$ is easy to explain: every counterexample presented to our learning algorithm consists of a point $x \in \mathbb{R}^d$ together with the real value $T(x)$. Together, these $d+1$ real numbers form a point in $\mathbb{R}^{d+1}$. Hence, the learning process may also be interpreted as receiving points in $\mathbb{R}^{d+1}$ (counterexamples) and trying to cover these points by a small number of hyperplanes (hypotheses).

Our main problem in learning $T(x)$ is to avoid getting large numbers of counterexamples in low-dimensional affine subspaces of $\mathbb{R}^{d+1}$. Suppose, we already received as counterexamples three points a, b and c that lie in this order on a straight line. Clearly, this line must be part of some f_j. If we receive another counterexample on this line, we cannot deduce any new information. Thus, we should avoid receiving four counterexamples that lie on a common line. As a second example, consider $3k$ counterexamples lying in a common plane. Such a plane need not be part of any f_j: In the worst case, there are k groups each consisting of 3 points on a line such that the intersections of the f_j with the plane exactly yield these lines. However, if there are $3k+1$ counterexamples lying in a common plane and *if no four of the counterexamples lie on a common line*, it can be checked that this plane must be part of some f_j. Hence, any line containing three counterexamples and any plane containing $3k+1$ counterexamples should be incorporated in the construction of our hypotheses. Similarly, for higher dimensional affine subspaces there are 'critical' numbers of points that force the subspace to be a subspace of some f_j (in case we simultanously avoid getting too many counterexamples in lower dimensional affine subspaces). It turns out that this critical number in a δ-dimensional affine subspace is exactly $\psi_k(\delta)$ as defined above.

Safe subsets are those subsets of the counterexamples that span affine subspaces containing a critical number of counterexamples; all safe subsets should be used in the construction of the next hypothesis. This is also the main idea of algorithm B_1 below. Algorithm B_1 maintains the following invariant: for any safe set P contained in the set of counterexamples received by the algorithm, which is consistent with some function T in $\mathcal{L}(d)_k^{\max}$, $S(P)$ is contained in the graph of one of the linear functions defining T.

The Algorithm $\mathbf{B}_1$. Let $P_{s-1} = \{(x_1, T(x_1)), (x_2, T(x_2)), \ldots, (x_{s-1}, T(x_{s-1}))\} \subset \mathbb{R}^{d+1}$ be the set of counterexamples known in the beginning of Step s. Let

$$P_{s-1}^{(1)}, P_{s-1}^{(2)}, \ldots$$

be an enumeration of the safe subsets of P_{s-1}. For $x \in \mathbb{R}^d$, an affine subspace $S(P_{s-1}^{(i)})$ of $\mathbb{R}^{d+1}$ assigns a $(d+1)$st coordinate to x iff the projection of the subspace into $\mathbb{R}^d$ contains the point x. Let $Coord_{s-1}(x)$ denote the set containing the $(d+1)$st coordinates assigned to x by the $S(P_{s-1}^{(i)})$ for all i. In other words,

$$Coord_{s-1}(x) = \{u : \exists i, (x_1, ..., x_d, u) \in S(P_{s-1}^{(i)})\}.$$

Then the hypothesis h_s of algorithm B_1 is defined by

$$h_s(x) := \begin{cases} \max Coord_{s-1}(x) & \text{if } Coord_{s-1}(x) \text{ is not empty} \\ 0 & \text{otherwise.} \end{cases}$$

LEMMA 14 *For all $s > 0$, and for every safe subset $P_{s-1}^{(i)}$, (i) there exists some component f_j of T such that $S(P_{s-1}^{(i)}) \subseteq f_j$, and (ii) $|P_{s-1}^{(i)}| = \psi_k(\delta(P_{s-1}^{(i)}))$.*

Proof: Assume the statement does not hold, and consider the first Step s in which a contradiction occurs. We consider the smallest dimensional $P = P_{s-1}^{(i)}$ that does not fulfill condition (i) or condition (ii). We distinguish three cases depending on the dimension of P and on whether condition (i) or (ii) is violated.

Suppose, $\delta(P) \geq 1$ and P does not fulfill (i). Then we consider the intersections $f_1 \cap P, \dots, f_k \cap P$. Each of these intersections is at most $\delta(P) - 1$ dimensional and hence contains at most $\psi_k(\delta(P) - 1)$ of the counterexamples. Consequently, P would contain at most $k\psi_k(\delta(P) - 1) = \psi_k(\delta(P)) - 1$ counterexamples and could not be safe.

Next suppose, $\delta(P) \geq 1$ and P fulfills (i) but not (ii). In the preceding Step $(s-1)$, all safe subsets fulfilled (i). Therefore, subspaces spanned by safe subsets agreed with the target function and could not receive any further counterexamples. Since $|P|$ was equal to $\psi_k(\delta(P))$ in Step $(s-1)$, it is also equal to $\psi_k(\delta(P))$ in Step s.

Finally, for $\delta(P) = 0$, the statement trivially holds. ■

THEOREM 12 *For $d, k \geq 1$, LC-ARB$(\mathcal{L}(d)_k^{\max}) \leq k\psi_k(d) \in O(k^d)$. Moreover, the hypotheses used in the learning algorithm consist of at most $O(k^{d^2+2d})$ linear pieces, and the learning algorithm is computationally feasible.*

Proof: By Lemma 14, every f_i contains at most $\psi_k(d)$ counterexamples. Hence, B_1 must terminate after at most $k\psi_k(d)$ counterexamples.

To get the bound on the combinatorial complexity of the hypotheses, we will use the notion of the VC-dimension of set systems: A set X is *shattered* by a set system $\mathcal{S}$, if all $2^{|X|}$ subsets of X can be obtained by intersecting X with some set in $\mathcal{S}$. The *VC-dimension* of a set system $\mathcal{S}$ is defined to be the cardinality of the largest set shattered by $\mathcal{S}$ (in case no largest shattered set exists, the VC-dimension is infinite). Sauer (1972) proved that a set system on p points with VC-dimension v consists of at most $O(p^v)$ sets.

We argue that the VC-dimension of the set system induced by all affine linear subspaces of $\mathbb{R}^{d+1}$ is at most $d + 2$: A set P^{VC} of points shattered by this system can only contain extreme points (otherwise, some non-extreme point is contained in a simplex formed by several other points in P^{VC}. The set consisting of the corners of this simplex cannot be obtained by intersecting P^{VC} with an affine subspace without containing this non-extreme point at the same time). If we have $d+3$ extreme points in $(d+1)$-dimensional space, the affine hull of any $d+2$ of them is already all of $S(P^{VC})$. Then this spanning set cannot be represented by intersecting P^{VC} with an affine subspace without containing the other point at the same time.

Finally, we take the set of counterexamples as a ground set and consider the set system induced by the affine subspaces used in the construction of the hypothesis. There are at most $O(k^d)$ counterexamples and the VC-dimension of the system is at most $d+2$. Now by Sauer's result the claimed result follows. ■

For $d = 1$, we can strengthen Theorem 12 to the following optimal result.

THEOREM 13 *For $k \geq 1$,* LC-ARB$(\mathcal{L}(1)_k^{\max}) = 3k - 1$.

Proof: The upper bound is obvious for $k = 1$. For $k \geq 2$, $\psi_k(1) = 3$ holds and the result in Theorem 12 yields LC-ARB$(\mathcal{L}(1)_k^{\max}) \leq 3k$. We can do better by observing that after $3k-1$ counterexamples, these points determine $k-1$ lines exactly (with each line containing three points) and leave two points for the last line. This yields the upper bound (with a computationally feasible learning algorithm).

For the lower bound, we will give an adversary argument. To this end, we define so-called *bad* point sets BPS$_j$, $1 \leq j \leq k$ fulfilling the following two properties.

(i) BPS$_j$ consists of $3j-1$ points in convex position. Let $p_1, \ldots, p_{3j-1}$ denote the points in BPS$_j$ sorted by increasing x-coordinate.

(ii) There exists an index $a \geq 0$ divisible by three such that BPS$_j$ is divided into three parts $L_j = \{p_1, \ldots, p_a\}$, $M_j = \{p_{a+1}, p_{a+2}\}$ and $R_j = \{p_{a+3}, \ldots, p_{3j-1}\}$. Scanning through L_j from left to right, every triple p_{3i-2}, p_{3i-1}, p_{3i} lies on a common line segment. Similarly, in R_j every triple p_{3i-3}, p_{3i-2}, p_{3i-1} lies on a common line segment. Note that L_j or R_j might be empty.

We claim that a malicious adversary can confront the learner with bad sets of counterexamples BPS$_1, \ldots,$ BPS$_k$. This yields $3k-1$ counterexamples and proves the lower bound.

For $k = 1$, the claim trivially holds, and it remains to show how to reach a BPS$_{j+1}$ from some BPS$_j$. We assume that neither L_j nor R_j is empty; otherwise analogous arguments will work. Thus, we must construct three legal counterexamples q_1, q_2 and q_3, no matter how the learner chooses his hypotheses.

The first counterexample is given somewhere between p_{a+1} and p_{a+2}, below the line thru p_{a+1} and p_{a+2} but above the lines thru p_a, p_{a+1} and p_{a+2}, p_{a+3}. There is ample space to place the new point q_1 such that the convexity condition still holds.

The x-coordinate of the second counterexample q_2 is chosen very close to the x-coordinate of q_1. The y-coordinate of q_2 is determined such that q_2 either lies on the line thru p_{a+1}, q_1 or on the line thru q_1, p_{a+2}. The actual choice depends on the learner's hypothesis; he can assign to any x-coordinate at most one y-coordinate, and we present to him the other point as new counterexample. (If the x-coordinate of q_2 lies to the left of q_1 and to the right of the intersection of the lines thru p_a, p_{a+1} and thru q, p_{a+2}, the new point q_2 does not violate the convexity condition).

Depending on the choice of q_2, either the triple p_{a+1}, q_2, q_1 is added to L_j or the triple q_2, q_1, q_{a+2} is added to R_j (since these points lie on a common line segment). In any case, we are left with a single point between L_j and R_j. Counterexample q_3 is put somewhere close to this single point such that it invalidates the next hypothesis and does not violate the convexity condition.

Obviously, BPS_j together with q_1, q_2 and q_3 forms a new bad point set BPS_{j+1}. We keep on presenting counterexamples till we reach BPS_k. The corresponding hidden function consists of the maximum of all $k-1$ lines going thru the triples in L_k and R_k together with the line thru the two points in M_k. ■

4.2. The maximum of a fixed number of linear functions

Next, we describe an algorithm for learning the maximum of a fixed number of linear functions. Our algorithm will be obtained by reducing the problem of learning $\mathcal{L}(d)_k^{\max}$ to the problem of learning unions of k subspaces in d dimensions.

For all positive integers d, k, define $\text{HYPER}_{d,k}$ to be the set of all $f : \mathbb{R}^d \rightarrow \{0,1\}$ such that there exist $\vec{a}_1, \ldots, \vec{a}_k \in \mathbb{R}^d, b_1, \ldots, b_k \in \mathbb{R}$ such that for all $\vec{x} \in \mathbb{R}^d$,

$$f(\vec{x}) = 1 \Leftrightarrow \exists j \leq k, \vec{a}_j \cdot \vec{x} = b_j.$$

The following lemma is a straightforward consequence of a result of (Long and Warmuth, 1993).

Lemma 15 ((Long and Warmuth, 1993)) *Choose positive integers d and k. There is a learning algorithm B for $\mathit{HYPER}_{d,k}$ such that, for any learning process with target T, for each t, B's tth hypothesis h_t has $h_t(\vec{x}) \leq T(\vec{x})$ for all $\vec{x} \in \mathbb{R}^d$, and which makes at most $(d+1)^k$ mistakes.*

We apply this in the following.

Theorem 14 *For integers $d, k \geq 1$,* $\text{LC-ARB}(\mathcal{L}(d)_k^{\max}) \leq (d+2)^k$.

Proof: Choose integers $d, k \geq 1$. Consider the following computationally feasible algorithm B for learning $\mathcal{L}(d)_k^{\max}$ using as a subroutine an algorithm B' for learning $\text{HYPER}_{d+1,k}$ with the properties described in Lemma 15. B works by maintaining a simulated learning process for B'. In each round, B generates a hypothesis from the hypothesis of B', receives a counterexample, and then generates a counterexample for B from its counterexample.

Suppose h'_t is the tth hypothesis of B'. Then if for each $\vec{x} \in \mathbb{R}^d$,

$$\text{PROJ}(h'_t, \vec{x}) = \{u_{d+1} : \vec{u} \in \mathbb{R}^{d+1}, h'_t(\vec{u}) = 1, (u_1, ..., u_d) = \vec{x}\},$$

then the tth hypothesis h_t of B is defined by

$$h_t(\vec{x}) = \begin{cases} \sup \mathrm{PROJ}(h'_t, \vec{x}) & \text{if } \mathrm{PROJ}(h'_t, \vec{x}) \neq \emptyset \\ 0 & \text{otherwise} \end{cases} \tag{4.15}$$

If B receives $(\vec{x}_t, y_t)$ as a counterexample to h_t, then it passes $((x_{t,1}, ..., x_{t,d}, y_t), 1)$ to B'.

Suppose B is learning a target $T \in \mathcal{L}(d)_k^{\max}$ defined, for $\vec{a}_1, ..., \vec{a}_k \in \mathbb{R}^d, b_1, ..., b_k \in \mathbb{R}$, by

$$T(\vec{x}) = \max\{\vec{a}_j \cdot \vec{x} + b_j : j \leq k\}.$$

Then define $T' \in \mathrm{HYPER}_{d+1,k}$ by

$$T'(\vec{u}) = 1 \text{ iff } \exists j \leq k, \vec{a}_j \cdot (u_1, ..., u_d) + b_j = u_{d+1}.$$

Notice that for each $\vec{x} \in \mathbb{R}^d$,

$$T(\vec{x}) = \max\{r : T'(x_1, ..., x_d, r) = 1\}. \tag{4.16}$$

We claim that for each t, the tth counterexample $((x_{t,1}, ..., x_{t,d}, y_t), 1)$ passed to B' satisfies $h'_t((x_{t,1}, ..., x_{t,d}, y_t)) = 0$. If $\{\vec{u} \in \mathbb{R}^{d+1} : h'_t(\vec{u}) = 1, (u_1, ..., u_d) = \vec{x}\} = \emptyset$; this is trivial. Assume otherwise. Since, by Lemma 15,

$$\{\vec{u} : h'_t(\vec{u}) = 1\} \subseteq \{\vec{u} : T'(\vec{u}) = 1\},$$

we have

$$\begin{aligned} &\sup\{u_{d+1} : \vec{u} \in \mathbb{R}^{d+1}, h'_t(\vec{u}) = 1, (u_1, ..., u_d) = \vec{x}_t\} \\ &\qquad \leq \sup\{u_{d+1} : \vec{u} \in \mathbb{R}^{d+1}, T'(\vec{u}) = 1, (u_1, ..., u_d) = \vec{x}_t\}. \end{aligned}$$

By (4.16), $h(\vec{x}_t) \neq y_t = T(\vec{x}_t)$ then implies

$$\begin{aligned} &\sup\{u_{d+1} : \vec{u} \in \mathbb{R}^{d+1}, h'_t(\vec{u}) = 1, (u_1, ..., u_d) = \vec{x}_t\} \\ &\qquad < y_t = \max\{u_{d+1} : \vec{u} \in \mathbb{R}^{d+1}, T'(\vec{u}) = 1, (u_1, ..., u_d) = \vec{x}_t\}. \end{aligned}$$

Thus $h'_t(x_{t,1}, ..., x_{t,d}, y_t) = 0$, and $((x_{t,1}, ..., x_{t,d}, y_t), 1)$ is a valid counterexample for B' learning T'. Since t was chosen without loss of generality, B' makes at least as many mistakes learning T' as B does learning T. Applying Lemma 15 then yields the desired result. ■

5. Learning the Maximum of d-defined Functions

In this section we investigate learning maxima of functions. This is a natural generalization of the preceeding section and (among others) learning OR's of boolean functions. We describe a condition on a class $\mathcal{F}$ of functions of a single variable,

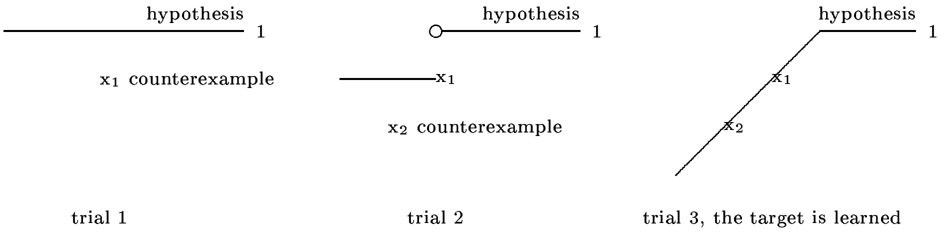

Figure 5. Learning a function $f_{a,b}$ from $\mathcal{E}$

such that the maximum of functions from this class is efficiently learnable. This condition is satisfied by a variety of natural function classes as indicated below.

Let $\mathcal{F} \subseteq Y^X$ be some function class where X and Y are totally ordered by $<$. In addition to $\mathcal{F}_k^{\max}$ we define the class $\mathcal{F}_k^{\mathrm{pmax}}$ of *piecewise* k-maxima. We call a k-maximum $f = \max_{1\leq i\leq k} f^i$ piecewise if there are disjoint intervals $I^1, \ldots, I^k$ with $X = \bigcup_{1\leq i\leq k} I^i$ such that $x \in I^i$ implies $f(x) = f^i(x)$. Informally this means that each function f^i appears only once in the graph of $\max_{1\leq i\leq k} f^i$.

Unfortunately it is not true that $\mathcal{F}_k^{\mathrm{pmax}}$ is easy to learn if $\mathcal{F}$ is easy to learn. Consider e.g. the class $\mathcal{E}$ of functions $f_{a,b} : \mathbb{R} \to \mathbb{R}$ with

$$f_{a,b}(x) = \begin{cases} 1 & \text{if } x \geq b \\ (x-a)/(b-a) & \text{if } x < b \end{cases}$$

for $a < b \in \mathbb{R}$. It is quite easy to see that LC-ARB$(\mathcal{E}) = 2$ (see Figure 5) but that LC-ARB$(\mathcal{E}_2^{\mathrm{pmax}}) = \infty$, since functions like

$$\begin{aligned} & g_{a,b}(x) = \\ & \max\{f_{a,b}(x), f_{0,2}(x)\} \\ & = \begin{cases} 1 & \text{if } x \geq b \\ (x-a)/(b-a) & \text{if } 2a/(2+a-b) < x < b \\ x/2 & \text{if } x \leq 2a/(2+a-b), \end{cases} \end{aligned}$$

$b - 2 < a < b < 2$, are in $\mathcal{E}_2^{\mathrm{pmax}}$ (see Figure 6 and compare with Example 1). Therefore we need a stronger restriction on the function class $\mathcal{F}$ than LC-ARB$(\mathcal{F}) < \infty$ to bound LC-ARB$(\mathcal{F}_k^{\mathrm{pmax}})$. A property which suffices is the notion of a *d-defined* function class, where a function class $\mathcal{F}$ is d-defined if any two functions $f^1, f^2 \in \mathcal{F}$ intersect in at most d points, i.e. $|\{x \in X : f^1(x) = f^2(x)\}| > d$ implies $f^1 = f^2$. Examples of d-defined classes are among others the class $\mathcal{P}_d$ of polynomials of degree at most d, the class of polynomials with at most $\lfloor (d-1)/4 \rfloor$ terms ("sparse polynomials"), quotients of sparse polynomials, and functions of the form $\sum_{i=1}^k a_i\sigma(b_i x + c_i)$ for arbitrary $a_i, b_i, c_i \in \mathbb{R}$, and bounded $k \in \mathbb{N}$ (where $\sigma(y) := 1/(1 + \mathrm{e}^{-y})$ is the widely studied *sigmoid activation function* in neural

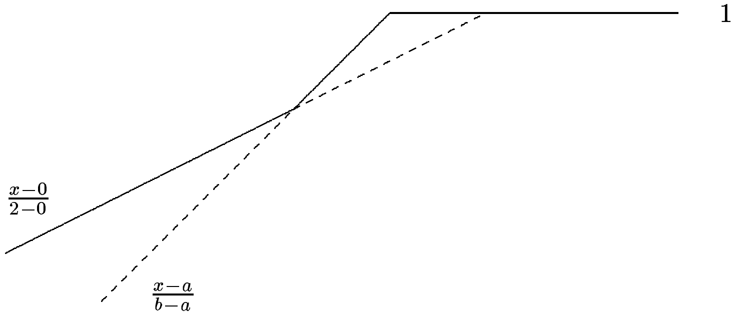

Figure 6. $g_{a,b} = \max\{f_{a,b}, f_{0,2}\}$. The class $\{g_{a,b} : b-2 < a < b < 2\}$ is not learnable.

networks). The d-definedness of these classes (for suitable $d \in \mathbb{N}$) follows from the fact that any polynomial with m terms has at most $2m+1$ zeros (by (Uspensky, 1948, p. 121)). Functions $\sum_{i=1}^{k} a_i\sigma(b_i x + c_i)$ can be rewritten as "rational" functions $\sum_{i=1}^{k} \frac{a_i}{1+C_i y^{b_i}}$ where $C_i = e^{-c_i}$ and $y = e^{-x}$. Observe that b_i is not integer but that the result of (Uspensky, 1948) can be generalized to this case. For the class of quotients of sparse polynomials a suitable d is $4kl + 1$ when nominator and denominator have at most k and l terms, respectively, and for sums of k sigmoids a suitable d is $4k2^{2k-1} + 1$.

Note that in contrast to Section 4 the here considered function class $\mathcal{F}_k^{\text{pmax}}$ may also include *non-convex* functions, and in contrast to (Kimber and Long, 1992) the norm of the derivative of functions in this class need not be bounded.

THEOREM 15 *If $\mathcal{F}$ is d-defined with $d \geq 1$ then* LC-ARB$(\mathcal{F}_k^{\text{pmax}}) = O(dk)$.

Remark. The above result is optimal up to constants insofar as for polynomials we can prove that LC-ARB$((\mathcal{P}_d)_k^{\text{pmax}}) = \Omega(dk)$. This can be seen by considering functions

$$\max\{0, \sum_{j=0}^{d} a_{ij}(x-2i)^{2j} : i = 1, \ldots, k\}$$

where for all i $a_{id} = -1$ and $\sum_{j=0}^{d-1} a_{ij} \leq 1$. The coefficients a_{ij} must be learned independently for each i since $\sum_{j=0}^{d} a_{ij}(x-2i)^{2j} \geq 0$ only if $|x - 2i| \leq 1$, see Figure 7.

Remark. A learning result for $\mathcal{F}_k^{\text{max}}$ is given in Section 5.3. The learning algorithms for both $\mathcal{F}_k^{\text{pmax}}$ and $\mathcal{F}_k^{\text{max}}$ are computationally feasible, see Remark 5.1.

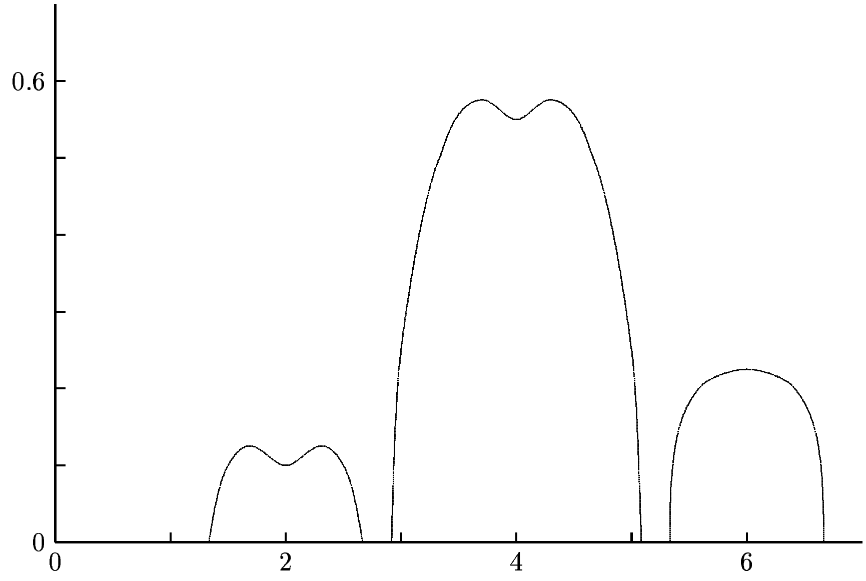

Figure 7. An example from $(\mathcal{P}_4)_7^{\mathrm{pmax}}$.

At first glance the proof of Theorem 15 seems easy because one could argue that the learner just has to get $d + 1$ examples for each piece of the target. But the problem is to determine which of the examples belong to which piece of the target function, i.e. how to partition the examples into subsets such that each subset belongs to one piece of the target. Thus the main point of the proof of Theorem 15 is to construct an algorithm which partitions the counterexamples and constructs its hypotheses in such a way that any piece of the target does not receive more than $O(d)$ counterexamples on its graph.

We achieve this goal by constructing an algorithm which only adds a new piece to its hypothesis if there is strong evidence that this piece is part of the target. The algorithm also maintains a set of cutting points to partition the set of counterexamples. These cutting points are also used to protect pieces which are very likely to be in the target against overlapping pieces which are less likely.

5.1. The Algorithm

Denote by E_s the set of X-coordinates of the counterexamples up to and including the s-th counterexample and denote by $y(x)$ the corresponding Y-value of the counterexample if $x \in E_s$. Since the algorithm always constructs hypotheses consistent with the previously seen counterexamples we have $|E_s| = s$. Some of the

$x \in E_s$ are used to cut X into intervals. We denote the set of these "cutting points" by $C_s \subseteq E_s$. (The algorithm will maintain the set C_s dynamically.) Furthermore denote by $x_1 < \ldots < x_s$ the sorted elements of E_s. For notational convenience we set $x_0 = -\infty$, $x_{s+1} = +\infty$.

At first the algorithm looks for all functions from $\mathcal{F}$ consistent with at least $2d+1$ consecutive counterexamples (this is "strong evidence" that these functions build pieces of the target). Let

$$\begin{aligned} Q_s = \{q : 1 \leq q \leq s - 2d \wedge \exists f_s^q \in \mathcal{F} \;\; \forall j = 0, \ldots, 2d \;\; (f_s^q(x_{q+j}) = y(x_{q+j}) \\ \wedge x_{q+j} \notin C_s)\} \end{aligned}$$

where the second condition states that none of the $2d+1$ counterexamples are cutting points (cutting points are not considered as regular counterexamples and do not give "evidence"). If $q \in Q_s$ we denote the appropriate function by f_s^q. For each $q \in Q_s$ let $D_s^q = (L_s^q, R_s^q) \subseteq X$ be the maximal interval around x_q which contains no cutting points and no inconsistent counterexamples for f_s^q,

$$L_s^q = \max\{x < x_q : x \in E_s, f_s^q(x) \neq y(x) \vee x \in C_s\},$$

$$R_s^q = \min\{x > x_q : x \in E_s, f_s^q(x) \neq y(x) \vee x \in C_s\},$$

such that the "range" D_s^q of a function/piece f_s^q is bounded by inconsistent counterexamples or cutting points and contains at least $2d+1$ counterexamples consistent with f_s^q. (As usual we define $\max\emptyset = -\infty$, $\min\emptyset = +\infty$.) Observe that each $x \in X$ is element of at most two distinct ranges: Assume $x \in D_s^1 \cap D_s^2 \cap D_s^3$, $L_s^1 \leq L_s^2 \leq L_s^3$. If $R_s^3 \leq R_s^2$ then $D_s^3 \subseteq D_s^2$ which by the d-definedness of $\mathcal{F}$ implies $f_s^2 = f_s^3$ and therefore $D_s^2 = D_s^3$. If $R_s^2 \leq R_s^3$ then $D_s^2 \subseteq D_s^1 \cup D_s^3$ and the number of counterexamples in one of $D_s^2 \cap D_s^1$ or $D_s^2 \cap D_s^3$ is at least $d+1$, which again implies $f_s^2 = f_s^1$, $D_s^2 = D_s^1$, or $f_s^2 = f_s^3$, $D_s^2 = D_s^3$, respectively.

The next hypothesis proposed by the algorithm is defined as the maximum of all pieces f_s^q,

$$H_s(x) = \begin{cases} y(x) & \text{, if } \; x \in E_s \\ \max\{f_s^q(x) : q \in Q_s \wedge x \in D_s^q\} & \text{, if } \; x \in \bigcup_{q \in Q_s} D_s^q \\ 0 & \; \text{otherwise.} \end{cases}$$

(Instead of 0 any element from Y may be used.)

If (x_s^e, y_s^e) is a new counterexample with $H_s(x_s^e) \neq y_s^e$ then E_s is updated as $E_{s+1} = E_s \cup \{x_s^e\}$ and the cutting points are updated according to the following rules U1 and U2:

U1: (There is a function f_s^q consistent with the new counterexample but this function has been overlapped by another function $f_s^{q'}$. Now the "correct" function is protected against the "wrong" function by introducing a new cutting point. Eventually the wrong function will not appear in the next hypothesis because its range becomes too small by the new cutting point.)

Assume

$$\exists q \in Q_s \ \ (x_s^e \in D_s^q \wedge \ f_s^q(x_s^e) = y_s^e). \tag{5.17}$$

Then there must be also some $q' \in Q_s$ with $x_s^e \in D_s^{q'}$, $H_s(x_s^e) = f_s^{q'}(x_s^e) > f_s^q(x_s^e) = y_s^e$. Since f_s^q seems to be the correct function the algorithm "protects" it against $f_s^{q'}$ by defining a new cutting point. If $L_s^{q'} < L_s^q$ then $C_{s+1} := C_s \cup \{L_s^q\}$. If $R_s^q < R_s^{q'}$ then $C_{s+1} := C_s \cup \{R_s^q\}$.

If $C_{s+1} = C_s \cup \{L_s^q\}$ this means that in the next hypothesis no function from the left can overlap f_s^q. From the definition of Q_s and the d-definedness of $\mathcal{F}$ one easily obtains that exactly one of the conditions $L_s^{q'} < L_s^q$, $R_s^q < R_s^{q'}$ is satisfied, since otherwise f_s^q and $f_s^{q'}$ would intersect in at least $2d+1$ points.

U2: Let F be the set of all functions $f \in \mathcal{F}$ with $f(x_s^e) = y_s^e$ for which there exists $0 \leq r_f \leq s - 2d$ for which $x_{r_f} < x_s^e < x_{r_f+2d+1}$, $f(x_{r_f}) \neq y(x_{r_f})$, $f(x_{r_f+1}) = y(x_{r_f+1}), \ldots, f(x_{r_f+2d}) = y(x_{r_f+2d})$, and $f(x_{r_f+2d+1}) \neq y(x_{r_f+2d+1})$. Then set $C_{s+1} := C_s \setminus \bigcup_{f \in F}\{x_{r_f+1}, \ldots, x_{r_f+2d}\}$. (We set for all $f \in \mathcal{F}$ that $f(-\infty) \neq y(-\infty)$, $f(+\infty) \neq y(+\infty)$).

The second update U2 is a more technical update rule. It removes some cutting points if there is evidence that they are misleading, partitioning the counterexamples into too many subsets. Evidence is given by the set F of candidates for new pieces in the next hypothesis, consistent with exactly $2d$ consecutive counterexamples in E_s and also consistent with the new counterexample (x_s^e, y_s^e). The update U2 enforces that the functions in F appear in the next hypothesis. Observe that the set F of U2 is empty if condition (5.17) is satisfied: With the notation of U1 and assuming $L_s^{q'} < L_s^q$ we have $|(L_s^q, R_s^{q'}) \cap E_s| \leq d$ (otherwise $f_s^q = f_s^{q'}$). Thus $|(L_s^{q'}, x_s^e) \cap E_s| \geq d+1$ and $|(x_s^e, R_s^q) \cap E_s| \geq d+1$. Let $f \in F$ and $D_f = \{x_{r_f+1}, \ldots, x_{r_f+2d}\}$. Then $|D_f \cap D_s^{q'}| \geq d+1$ or $|D_f \cap D_s^q| \geq d$ which implies that f coincides with $f_s^{q'}$ or f_s^q in at least $d+1$ points (observe that $f(x_s^e) = y(x_s^e) = y_s^q(x_s^e)$). Thus $f = f_s^{q'}$ or $f = f_s^q$ and therefore f is consistent with at least $2d+1$ consecutive counterexamples in E_s, which contradicts the definition of F.

Remark. Observe that the hypotheses of our algorithm consist of at most $O(dk)$ pieces of functions from $\mathcal{F}_2^{\text{pmax}}$. This follows from Theorem 15 which states that the algorithm receives at most $O(dk)$ counterexamples, and from the fact that each $x \in X$ is element of at most two distinct ranges D_s^q.

Concerning computational complexity, let the amount of time for computing the unique function consistent with some $d+1$ examples and calculating $f(x)$ for $f \in \mathcal{F}$, $x \in X$, be bounded by some constant $\mathrm{CC}_{\mathcal{F}}$. Then the computational complexity of our algorithm is bounded by some small polynomial in $d, k, \mathrm{CC}_{\mathcal{F}}$.

5.2. Analysis

We will argue that the above algorithm can receive at most $O(dk)$ counterexamples. Let $T = \max_{1 \le i \le k} T^i$ be the target function and let $I^1, \ldots, I^k$ be the corresponding intervals. To bound the number of counterexamples we will show that

$$s \;\le\; 5 \sum_{1 \le i \le k} \min\{\nu_s^i, 2d+1\} + |C_s|, \tag{5.18}$$

$$|C_s| \;\le\; s/(d+1)+1. \tag{5.19}$$

This implies $s \le [5k(2d+1)+1]/[1-1/(d+1)] = O(dk)$ for $d \ge 1$.

At first we prove that there can be only few cutting points, showing that there are at least d counterexamples between two cutting points.

Lemma 16 *If x_{r_1}, x_{r_2} are counterexamples with $r_1 \ne r_2$ and $x_{r_1}, x_{r_2} \in C_s$ then $|r_1 - r_2| \ge d+1$.*

Proof: The proof is by induction on s. Since new cutting points are only generated by U1 we assume with the notations of U1 and w.l.o.g. that $L_s^{q'} < L_s^q$ and that L_s^q is the new cutting point. Since $|D_s^q \cap D_s^{q'} \cap E_s| \le d$ (otherwise $f_s^q = f_s^{q'}$) it follows that $|(L_s^{q'}, L_s^q] \cap E_s| \ge d+1$. Since $|(L_s^q, R_s^q) \cap E_s| \ge 2d+1$ the lemma follows from $C_s \cap (D_s^q \cup D_s^{q'}) = \emptyset$. ■

Clearly Lemma 16 implies (5.19).

Corollary 1 U2 *deletes at most 4 elements from C_s.*

Proof: Since for all $f \in F$ $x_{r_f} < x_s^e < x_{r_f+2d+1}$ we have

$$|\bigcup_{f \in F} \{x_{r_f+1}, \ldots, x_{r_f+2d}\}| \le 4d$$

which together with Lemma 16 implies the corollary. ■

For proving (5.18) we start showing that there are no cutting points in I^i if $\nu_s^i \ge 2d+1$.

Lemma 17 *If $x \in E_s \cap I^i$ and $\nu_s^i \ge 2d+1$ then $x \notin C_s$.*

Proof: To prove the lemma we have to introduce some notation. Let $\bar{D}_s^i = (\bar{L}_s^i, \bar{R}_s^i)$ be the maximal interval where T^i is consistent with the counterexamples seen so far,

$$\bar{L}_s^i = \max\{x < I^i : x \in E_s, y(x) \ne T^i(x)\},$$

$$\bar{R}_s^i = \min\{x > I^i : x \in E_s, y(x) \ne T^i(x)\},$$

($x < I^i$ has the obvious interpretation). Furthermore let $\bar{\nu}^i_s = |\bar{D}^i_s \cap E_s|$ be the number of counterexamples in this interval. Clearly $I^i \subseteq \bar{D}^i_{s+1} \subseteq \bar{D}^i_s$ and $\nu^i_s \leq \bar{\nu}^i_s$. Thus it is sufficient to prove that $\bar{\nu}^i_s \geq 2d+1$ implies $\bar{D}^i_s \cap C_s = \emptyset$.

The proof is by induction on s. Since $\bar{\nu}^i_s$ increases by at most 1 if s increases by 1, we first prove that $\bar{\nu}^i_s = 2d$ and $\bar{\nu}^i_{s+1} = 2d+1$ implies $\bar{D}^i_{s+1} \cap C_{s+1} = \emptyset$. If $\bar{\nu}^i_s = 2d$ and $\bar{\nu}^i_{s+1} = 2d+1$ then $x^e_s \in \bar{D}^i_s$ and $\bar{D}^i_s = \bar{D}^i_{s+1}$. Furthermore $T^i \in F$ as can be seen by checking the conditions of U2. Thus all cutting points are removed from $\bar{D}^i_s$.

It remains to prove that if $\bar{\nu}^i_s \geq 2d+1$ and $\bar{\nu}^i_{s+1} \geq 2d+1$ then no cutting point is generated in $\bar{D}^i_{s+1}$. Since new cutting points are only generated by U1 we assume w.l.o.g. that with the notation of U1 $L^{q'}_s < L^q_s < x^e_s < R^{q'}_s < R^q_s$ and L^q_s is the new cutting point. Now assume that $L^q_s \in \bar{D}^i_{s+1}$. Since by the d-definedness of $\mathcal{F}$ $|(L^q_s, R^{q'}_s) \cap E_s| \leq d$ (otherwise $f^q_s = f^{q'}_s$) we get $|(L^{q'}_s, L^q_s] \cap E_s| \geq d+1$. Thus $|(L^{q'}_s, L^q_s] \cap \bar{D}^i_s \cap E_s| \geq d+1$ or $|(L^q_s, R^q_s) \cap \bar{D}^i_s \cap E_s| \geq d+1$ which implies $T^i = f^{q'}_s$ or $T^i = f^q_s$. Since $T^i = f^{q'}_s$ yields the contradiction $T(x^e_s) \geq T_i(x^e_s) = f^{q'}_s(x^e_s) > f^q_s(x^e_s) = y(x^e_s)$ we have $T^i = f^q_s$. Now $L^q_s \in \bar{D}^i_{s+1} \subseteq \bar{D}^i_s$ implies $\L^q_s \notin C_s$ (by the induction hypothesis) and by the definition of L^q_s $y(L^q_s) \neq f^q_s(L^q_s)$ which contradicts $f^q_s(L^q_s) = T^i(L^q_s) = T(L^q_s)$ and proves the lemma. ■

Now (5.18) follows by induction on s from the lemma below since $\nu^i_0 = 0$ and $C_0 = \emptyset$.

LEMMA 18 $5\sum_{1\leq i\leq k} \min\{\nu^i_{s+1}, 2d+1\} + |C_{s+1}| \geq 5\sum_{1\leq i\leq k} \min\{\nu^i_s, 2d+1\} + |C_s| + 1$.

Proof: If $x^e_s \in I^i$ and $\nu^i_s < 2d+1$ the statement holds by Corollary 1. If $\nu^i_s \geq 2d+1$ then by Lemma 17 there exists $q \in Q_s$ with $f^q_s = T^i$ and $x^e_s \in D^q_s$. Since $y^e_s = T^i(x^e_s) = f^q_s(x^e_s)$ condition (5.17) is satisfied and a new cutting point is generated, thus $|C_{s+1}| = |C_s| + 1$, which gives the lemma. ■

5.3. Learning $\mathcal{F}^{\max}_k$

THEOREM 16 *If $\mathcal{F}$ is d-defined with $d \geq 1$ then* LC-ARB$(\mathcal{F}^{\max}_k) = O(dk^2)$.

Let again $T = \max_{1\leq i\leq k} T^i$ be the target and E_s the set of counterexamples. Furthermore let $\mu^i_s = |\{x \in E_s : y(x) = T^i(x)\}|$ be the number of counterexamples consistent with T^i. The following lemma will prove useful in constructing a computationally feasible learning algorithm for $\mathcal{F}^{\max}_k$. It states that any function consistent with at least $(d+1)k$ counterexamples appears in the target.

LEMMA 19 *For all $f \in \mathcal{F}$ with $|\{x \in E_s : f(x) = y(x)\}| \geq (d+1)k$ there exists a T^i with $f = T^i$.*

Proof: Since there must be a T_i with $|\{x \in E_s : f(x) = y(x) = T_i(x)\}| \geq d+1$ the lemma follows from the d-definedness of $\mathcal{F}$. ■

Now the hypothesis of the learning algorithm is constructed as follows. Let

$$F_s = \{f \in \mathcal{F} : |\{x \in E_s : f(x) = y(x)\}| \geq (d+1)k\}$$

and

$$H_s(x) = \begin{cases} y(x) & \text{, if } \ x \in E_s \\ max_{f \in F_s} f(x) & \text{, otherwise (if } F_s = \emptyset \text{ any } y \in Y \text{ will do).} \end{cases}$$

Theorem 16 follows from the following lemma which states that each T^i will receive at most $(d+1)k$ counterexamples.

LEMMA 20 $\forall i = 1, \ldots, k \quad \mu_s^i \leq (d+1)k.$

Proof: Assume $\mu_s^i = (d+1)k$ and $\mu_{s+1}^i > (d+1)k$. Then $T^i \in F_s$ and $y(x_s^e) = T^i(x_s^e)$. Thus $y(x_s^e) < H_s(x_s^e)$ which yields the contradiction $y(x_s^e) < T(x_s^e)$ since by Lemma 19 $H_s(x) \leq T(x)$ for all $x \in X$. ■

6. Conclusion

We have shown in this paper a number of general structural properties for the complexity of learning functions with a larger range than $\{0, 1\}$ in the most common non-stochastic models for on-line learning. One somewhat surprising structural symmetry that has emerged in section 2 is the "dual" definition of the learning complexity for arbitrary function classes. We have exhibited in Theorems 1, 4, and 5 (with the help of adversary trees) purely combinatorial "max-min" definitions of the learning complexity of a function class, which turns out to be equivalent to the usual "min-max" definition of the learning complexity in terms of the performance of learning algorithms for the respective learning-"game". On the side these equivalences show that it suffices to consider in these worst case models for on-line function learning only adversaries of a relatively simple nature.

In section 3 we have investigated another general structural property of the complexity of function learning: the relationship between the learning complexity of a complex function class in terms of the learning complexity of its parts. We have been able to give optimal upper and lower bounds, and we have exhibited a universal constant $1/\log(4/3)$ that plays an important role in these relationships.

In sections 4 and 5 we have substantially enlarged the pool of interesting function classes for which positive learning results can be achieved in a worst case model for on-line function learning. Theorem 12 provides a computationally feasible learning algorithm for convex piecewise linear functions over $\mathbb{R}^d$, thereby breaking the barrier of exhibiting larger "learnable" classes of functions of several variables than the standard example of linear functions. In section 5 we have investigated in a systematic manner the conditions for achieving a positive learning result for the class of functions that are pointwise maxima of arbitrary finite sets of functions from a certain class $\mathcal{F}$. In particular we have exhibited a general structural property ("d-definedness") for function classes $\mathcal{F}$, which provides a sufficient condition for proving a positive learning result for pointwise maxima of functions from $\mathcal{F}$. As a special case we get the first positive learning results for the classes of pointwise maxima of polynomials of bounded degree, of sparse polynomials, and of linear combinations of a fixed number of sigmoidal functions over a single variable.

Acknowledgements

We thank Yoav Freund, Wolfgang Ring and Hans Ulrich Simon for valuable conversations. We also thank two anonymous referees for their helpful comments, including for pointing out an error in an earlier version of this paper. Phil Long was supported by a Lise Meitner Fellowship from the Fonds zur Förderung der wissenschaftlichen Forschung (Austria), and AFOSR grant F49620–92–J–0515. His present address is: Computer Science Department, Duke University, Box 90129, Durham, NC 27708 USA. Gerhard Woeginger is supported by the Christian Doppler Laboratorium für Diskrete Optimierung. His present address is: Institut für Mathematik, TU Graz, Kopernikusgasse 24, A-8010 Graz, Austria.

Notes

1. Some lower bounds require that $|Y| \geq 2$. The case $|Y| \leq 1$ is trivial.
2. This is in contrast to the case of probabilistic models of learning, where efficient algorithms with good learning performance have been discovered for this function class (Kearns, Schapire and Sellie, 1992).

References

Auer, P., and Long, P.M. (1994). Simulating access to hidden information while learning. *Proceedings of the 26th Annual ACM Symposium on the Theory of Computing.*

Auer, P., Long, P.M., Maass, W., and Woeginger, G.J. (1993). On the complexity of function learning. *Proceedings of the Sixth Annual ACM Conference on Computational Learning Theory*, pages 392–401.

Angluin, D. (1988). Queries and concept learning. *Machine Learning*, 2(4):319–342.

Barland, I. (1992). Some ideas on learning with directional feedback. Master's thesis, Computer Science Department, UC Santa Cruz.

Berlekamp, E.R. (1968). Block coding for the binary symmetric channel with noiseless, delayless feedback. In *Error Correcting Codes.* New York: Wiley.

Barzdin, J.M., and Frievald, R.V. (1972). On the prediction of general recursive functions. *Soviet Math. Doklady*, 13:1224–1228.

Cesa-Bianchi, N., Freund, Y., Helmbold, D.P., and Warmuth, M.K. (1994). On-line prediction and conversion strategies. In *Proceedings of the First Euro-COLT Workshop.* The Institute of Mathematics and its Applications, to appear.

Cesa-Bianchi, N., Long, P.M., and Warmuth, M.K. (1993). Worst-case quadratic loss bounds for a generalization of the Widrow-Hoff rule. In *Proceedings of the 6th Annual Workshop on Comput. Learning Theory.*

Dawid, A. (1984). Statistical theory: The prequential approach. *Journal of the Royal Statistical Society (Series A)*, pages 278–292.

Faber, V., and Mycielski, J. (1991). Applications of learning theorems. *Fundamenta Informaticae*, 15(2):145–167.

Feder, M., Merhav, N., and Gutman, M. (1992). Universal prediction of individual sequences. *IEEE Transactions of Information Theory*, 38:1258–1270.

Kimber, D. and Long, P.M. (1992). The learning complexity of smooth functions of a single variable. In *Proc. 5th Annu. Workshop on Comput. Learning Theory.*

Kearns, M.J., Schapire, R.E., and Sellie, L.M. (1992). Toward efficient agnostic learning. In *Proc. 5th Annu. Workshop on Comput. Learning Theory.*

Littlestone, N. (1988). Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Machine Learning*, 2:285–318.

Littlestone, N. (1989). *Mistake Bounds and Logarithmic Linear-threshold Learning Algorithms* PhD thesis, Technical Report UCSC-CRL-89-11, University of California Santa Cruz.

Littlestone, N., Long, P.M., and Warmuth, M.K. (1991). On-line learning of linear functions. In *Proc. of the 23rd Symposium on Theory of Computing.*

Littlestone, N., and Warmuth, M.K. (1991). The weighted majority algorithm. Technical Report UCSC-CRL-91-28, UC Santa Cruz. A preliminary version appeared in *the Proceedings of the 30th Annual IEEE Symposium of the Foundations of Computer Science.*

Long, P.M., and Warmuth, M.K. (in press). Composite geometric concepts and polynomial predictability. *Inform. Comput.*.

Maass, W. (1991). On-line learning with an oblivious environment and the power of randomization. *Proc. 4th Annu. Workshop on Comput. Learning Theory.*

Maass, W., and Turán, G. (1992). Lower bound methods and separation results for on-line learning models. *Machine Learning*, 9:107–145.

Mycielski, J. (1988). A learning algorithm for linear operators. *Proceedings of the American Mathematical Society*, 103(2):547–550.

Rivest, R.L., Meyer, A.R., Kleitman, D.J., Winklmann, K., and Spencer, J. (1980). Coping woth errors in binary search procedures. *Journal of Computer and System Sciences*, 20:396–404.

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

Spencer, J. (1992). Ulam's searching game with a fixed number of lies. *Theoretical Computer Science*, 95(2):307–321.

Uspensky, J.V. (1948). *Theory of Equations.* McGraw-Hill.

Vovk, V. (1990). Aggregating strategies. In *Proc. 3rd Annu. Workshop on Comput. Learning Theory.*

Vovk, V. (1992). Universal forecasting algorithms. *Inform. Comput.*, 96(2):245–277.