A Complete and Tight Average-Case Analysis of Learning Monomials

Rüdiger Reischuk*
Institut für Theoretische Informatik
Med. Universität zu Lübeck
Wallstraße 40
23560 Lübeck, Germany
reischuk@informatik.mu-luebeck.de

Thomas Zeugmann†
Department of Informatics
Kyushu University
Kasuga 816-8580
Japan
thomas@i.kyushu-u.ac.jp

September 1998

Abstract

We advocate to analyze the average complexity of learning problems. An appropriate framework for this purpose is introduced. Based on it we consider the problem of *learning monomials* and the special case of learning monotone monomials *in the limit* and for *on-line predictions* in two variants: from positive data only, and from positive and negative examples. The well-known *Wholist algorithm* is completely analyzed, in particular its average-case behavior with respect to the class of *binomial distributions.* We consider different complexity measures: the *number of mind changes, the number of prediction errors, and the total learning time.* Tight bounds are obtained implying that worst case bounds are too pessimistic. On the average learning can be achieved *exponentially faster.*

Furthermore, we study a new learning model, *stochastic finite learning*, in which, in contrast to PAC learning, some information about the underlying distribution is given and the goal is to find a *correct* (not only approximatively correct) hypothesis. We develop techniques to obtain good bounds for stochastic finite learning from a precise average case analysis of strategies for learning in the limit and illustrate our approach for the case of learning monomials.

1. Introduction

Learning concepts efficiently has attracted considerable attention during the last decade. There are several points of concern to be addressed when dealing with this problem. Research following the traditional lines of inductive inference has mainly considered the update time, that is the effort to compute a single new hypothesis after having seen one more example of the concept class. Starting with Valiant's pioneering paper [22], the total amount of

*Part of this work was performed while the author visited the Department of Informatics at Kyushu University and was supported by the Japan Society for the Promotion of Science under Grant JSPS 29716102.

†This author has been supported by the Grant-in-Aid for Scientific Research in Fundamental Areas from the Japanese Ministry of Education, Science, Sports, and Culture under grant no. 10558047.

time needed to solve a given learning problem has been investigated as well. The complexity bounds established within the PAC model are usually *worst-case* bounds. In experimental studies large gaps have often been observed between the time bounds obtained by a mathematical analysis and the actual runtime of a learner on typical data. This phenomenon can easily be explained. Data from running tests provide information about the *average-case* performance of a learner, rather than its worst-case behavior unless the number of tests is extremely large or test examples are drawn from very specific distributions that favor difficult inputs for the algorithm to be tested. Since algorithmic learning has a lot of practical applications it is of great interest to analyze the average-case performance, too, and to obtain tight bounds that say something about the typical behavior in practice.

Pazzani and Sarret [17] have proposed a framework for analyzing the average-case behavior of learning algorithms. Several authors have followed their approach (cf., e.g., [15, 16]). Their main goal is to predict the expected accuracy of the hypothesis produced with respect to the number of training examples. However, the results obtained so far are not satisfactory. Typically, the probability that an example uniformly drawn at random is misclassified by the current hypothesis is estimated by a complicated formula. The *evaluation* of this formula, however, as well as the computation of the corresponding expectation has been done by Monte-Carlo simulations. Clearly, such an approach does not provide general results about the average-case behavior for broader classes of distributions. Moreover, such an approach makes it hard – if not impossible – to compare the bounds obtained with those proved for the PAC model.

In this paper we outline a new setting to study the average-case behavior of learning algorithms that overcomes these drawbacks and illustrate it for the problem of learning monomials. In Section 3 we will introduce the learning models. Section 4 describes an algorithm for learning monomials. The average case behavior of this learner is then analyzed in the following sections.

2. Preliminaries

Let $\mathbb{N} = \{0, 1, 2, \ldots\}$ denote the set of all natural numbers, and let $\mathbb{N}^+ := \mathbb{N} \setminus \{0\}$. If M is a set $|M|$ will be used for its cardinality. For an infinite sequence d and $j \in \mathbb{N}^+$ let $\boldsymbol{d[j]}$ denote the initial segment of d of length j. $(0, 1)$ will be used to denote the real interval from 0 to 1 excluding both endpoints.

For $n \in \mathbb{N}^+$, let $\mathcal{X}_n = \{0,1\}^n$ be the **learning domain** and $\wp(\mathcal{X}_n)$ the power set of $\mathcal{X}_n$. A subset c of $\mathcal{X}_n$ will be called a **concept**, and a subset $\mathcal{C}$ of $\wp(\mathcal{X}_n)$ a **concept class**. The notation c will also be used to denote the characteristic function of a subset, that is for $b \in \mathcal{X}_n$: $c(b) = 1$ iff $b \in c$. To define the particular classes of concept we want to deal with in this paper let $\mathcal{L}_n = \{x_1, \bar{x}_1, x_2, \bar{x}_2 \ldots, x_n, \bar{x}_n\}$ be a set of literals. x_i is a *positive* literal and $\bar{x}_i$ a *negative* one. A conjunction of literals defines a **monomial**. For a monomial m let $\boldsymbol{\#(m)}$ denote its length, that is the number of literals in it.

m describes a subset $L(m)$ of $\mathcal{X}_n$, in other words a concept, in the obvious way: the concept contains exactly those binary vectors for which the monomial evaluates to 1, that is $L(m) := \{b \in \mathcal{X}_n \mid m(b) = 1\}$. The collection of objects we are going to learn is the set $\boldsymbol{\mathcal{C}_n}$ of all concepts that are describable by monomials over $\mathcal{X}_n$. There are two trivial concepts, the empty subset and $\mathcal{X}_n$ itself. $\mathcal{X}_n$, which will also be called "TRUE", can be represented by the empty monomial, while "FALSE", the empty concept, can be represented by, e.g., $x_1\bar{x}_1$.

Note that there is a slight difference between the class $\mathcal{C}_n$ of all concepts describable by monomials and the set of all monomials itself. The concept "FALSE" has several descriptions, for example $x_2\bar{x}_2$ or $x_1\bar{x}_1 \ldots x_n\bar{x}_n$ as well. However, it is the only one. In order to avoid ambiguity, we shall always represent "FALSE" by the monomial $x_1\bar{x}_1 \ldots x_n\bar{x}_n$. Therefore, in the following we often identify the set of all monomials over $\mathcal{L}_n$ and the concept class $\mathcal{C}_n$. Note that $|\mathcal{C}_n| = 3^n + 1$. We will also consider the subclass $\boldsymbol{\mathcal{CM}_n}$ of $\mathcal{C}_n$ consisting of those concepts that can be described by **monotone** monomials, i.e., by monomials containing positive literals only. It holds $|\mathcal{CM}_n| = 2^n$.

3. Learning Models and Complexity Measures

To define a learning model one has to specify additionally a machine model for the *learning algorithm*, the *source of information*, the *hypotheses spaces* used, and the *criteria of success*. A learning algorithm will simply be called *learner*.

The first learning model we are going to deal with is the ***on-line prediction*** model going back to Barzdin and Freivald [1] and Littlestone [13]. In this setting the source of information is specified as follows. The learner is given a sequence of labeled examples $d = \langle b_1,\, c(b_1),\, b_2,\, c(b_2),\, b_3,\, c(b_3),\, \ldots\rangle$ from the concept c, where the $b_j \in \mathcal{X}_n$, and $c(b_j) = 1$ if $b_j \in c$ and $c(b_j) = 0$ otherwise. The examples b_j are picked arbitrarily and the information provided is assumed to be without any errors. We refer to such sequences as ***data sequences*** and use $data(c)$ to denote the set of all data sequences for concept c.

In the on-line model the hypothesis space is not defined explicitly; it is left to the learner to choose an appropriate representation. For $j = 1, 2, 3, \ldots$, a learner P must predict $c(b_j)$ after having seen $d[2j-1] = \langle b_1,\, c(b_1), \ldots, b_{j-1},\, c(b_{j-1}), b_j\rangle$. We will denote this hypothesis by $P(d[2j-1])$. Then he receives the true value $c(b_j)$ and the next Boolean vector b_{j+1} . The learner has successfully learned if he eventually reaches a point beyond which it always predicts correctly. More formally, we define:

Definition 1. *A concept class* $\mathcal{C}$ *is called* ***on-line predictable*** *if there is a learner* P *such that for all concepts* $c \in \mathcal{C}$ *and all data sequences* $d = (d_j)_{j \in \mathbb{N}^+} \in data(c)$

(1) $P(d[2j-1])$ *is defined for all* j *, and*

(2) $P(d[2j-1]) = d_{2j}$ *for all but finitely many* j *.*

For on-line prediction, the ***complexity measure*** considered is the *number of prediction errors* made. Note that the prediction goal can always be achieved trivially if the learning domain is finite. Therefore, we aim to minimize the number of prediction errors when learning monomials.

Next, let us define Gold-style [6] ***learning in the limit***. One distinguishes between learning from positive and negative data and learning from positive data only. For a concept c, let $info(c)$ be the set of those data sequences $\langle b_1,\, c(b_1),\, b_2,\, c(b_2),\, b_3,\, c(b_3),\, \ldots\rangle$ in $data(c)$ that contain each element b of the learning domain $\mathcal{X}_n$ at least once. Such a sequence is called ***informant***. For ease of notation let us pair each element b_j with its classification $c(b_j)$. Then the j-th entry of an informant sequence d will be $d_j := (b_j, c(b_j))$.

A ***positive presentation*** of c is a data sequence that contains only elements of c and each one at least once. Thus all the values $c(b_j)$ are equal to 1 and thus could be omitted. In this case we will denote the sequence simply by $d = b_1, b_2, b_3, \ldots$. Let $d[j]^+ := \{b_i \mid 1 \le i \le j\}$ be the set of all examples contained in the prefix of d of length j, and let $pos(c)$ denote the set of all positive presentations of c. The elements of $pos(c)$ are also called a ***text*** for c.

A limit learner is an ***inductive inference machine*** (abbr. IIM). Such a machine M works as follows. As inputs it gets incrementally growing segments of a positive presentation (resp. of an informant) d. After each new input, it outputs a hypothesis $M(d[j])$ from a predefined hypothesis space $\mathcal{H}$. Each hypothesis refers to a unique element of the concept class. Note that the hypothesis space and the concept class can coincide.

DEFINITION 2. *Let $\mathcal{C}$ be a concept class and let $\mathcal{H}$ be a hypothesis space for it. $\mathcal{C}$ is called* ***learnable in the limit*** *from positive presentation (resp. from informant) if there is an IIM M such that for every $c \in \mathcal{C}$ and every $d \in pos(c)$ (resp. $d \in info(c)$),*

(1) *$M(d[j])$ is defined for all j,*

(2) *$M(d[j]) = h$ for all but finitely many j, where $h \in \mathcal{H}$ is a hypothesis referring to c.*

For the concept class $\mathcal{C}_n$ the hypothesis space $\mathcal{H}_n$ will be chosen as the set of all monomials over $\mathcal{L}_n$, whereas for $\mathcal{MC}_n$ it is the set of all monotone monomials. Again, learning in the limit of $\mathcal{C}_n$ and $\mathcal{MC}_n$ can always be achieved due to the finite learning domain. The question remains how efficiently this can be done.

The first complexity measure we will consider is the ***mind change*** complexity. A mind change occurs iff $M(d[j]) \neq M(d[j+1])$. Clearly, this measure is closely related to the number of prediction errors. Both complexity measures say little about the total amount of data and time needed until a concept is guessed correctly. Thus, for learning in the limit we will also measure the *time complexity.* Following Daley and Smith [4] we define the total learning time as follows. Let M be any IIM that learns a concept class $\mathcal{C}$ in the limit. Then, for $c \in \mathcal{C}$ and a text or informant d for c, let

$$\boldsymbol{Conv(M,d)} =_{df} \text{the least number } i \in \mathbb{N}^+ \text{ such that for all } j \ge i,\ M(d[j]) = M(d[i])$$

denote the ***stage of convergence*** of M on d. Moreover, by $T_M(d_j)$ we denote the number of steps to compute $M(d[j])$. We measure this quantity as a function of the length of the input and refer to it as the *update time.* Finally, the **total learning time** taken by the IIM M on a sequence d is defined as

$$\boldsymbol{TT(M,d)} \quad := \quad \sum_{j=1}^{Conv(M,d)} T_M(d[j]) \ .$$

Given a probability distribution D on the data sequences d we like to evaluate the *expectation* of $TT(M,d)$ with respect to D, the **average total learning time**.

4. The Wholist Algorithm Learning Monomials

In this section we present Haussler's Wholist algorithm for on-line prediction of monomials [8]. For learning in the limit this algorithm can be modified in a straightforward way. The limit learner computes a new hypothesis using only the most recent example received and his old hypothesis. Such learners are called *iterative* (cf. [3, 10]).

Let $c \in \mathcal{C}_n$ and $d = \langle b_1,\, c(b_1),\, b_2,\, c(b_2),\, b_3,\, \ldots,\, \rangle$ be a data sequence for c. Furthermore, let $b_i = b_i^1 b_i^2 \ldots b_i^n$ denote the i-th Boolean vector in d. Remember that a monomial h without any literals represents the concept "TRUE", that is $h(b) = 1$ for all $b \in \mathcal{X}_n$.

***Algorithm $\mathcal{P}$*:**

On input sequence $\langle b_1,\, c(b_1),\, b_2,\, c(b_2),\, \ldots\rangle$ do the following:

Initialize $h_0 := x_1\bar{x}_1 \ldots x_n\bar{x}_n$.
for $i = 1, 2, \ldots$ **do**
  let h_{i-1} denote $\mathcal{P}$'s internal prediction hypothesis produced before receiving b_i;
  when receiving b_i predict $h_{i-1}(b_i)$;
  read $c(b_i)$;
  if $h_{i-1}(b_i) = c(b_i)$ **then** $h_i := h_{i-1}$
    else for $j := 1$ **to** n **do**
      if $b_i^j = 1$ **then** delete $\bar{x}_j$ in h_{i-1} **else** delete x_j in h_{i-1};
    let h_i be the resulting monomial
end.

Before analyzing $\mathcal{P}$ we give an illustration with the following example. Let $n = 7$ and $c \in \mathcal{C}_n$ be described by $m = x_1\bar{x}_2x_4x_7$. Suppose the input data sequence starts as follows: $\langle 1001111,\, 1,\, 0110110,\, 0,\, 1011101,\, 1,\, 1011001,\, 1,\, \ldots\rangle$.

> Initially, $\mathcal{P}$ always predicts $h_0(1001111) = 0$. However, the true value is 1, and thus $\mathcal{P}$ executes the loop described above resulting in $h_1 = x_1\bar{x}_2\bar{x}_3x_4x_5x_6x_7$. After having read $b_2 = 0110110$, $\mathcal{P}$ predicts $h_1(0110110) = 0$, which is true. Hence, $h_2 = h_1$. Next, $\mathcal{P}$ reads 1011101 and predicts $h_2(1011101) = 0$, which is wrong. Thus, $\mathcal{P}$ executes the loop and computes $h_3 = x_1\bar{x}_2x_4x_5x_7$. Now, $\mathcal{P}$ reads 1011001 and predicts 0, which again is wrong. $\mathcal{P}$ executes the loop again generating $h_4 = x_1\bar{x}_2x_4x_7$. Since this internal hypothesis equals the target monomial, $\mathcal{P}$ makes no more prediction errors. Consequently, it does not change its internal hypothesis any further.

Note that the algorithm is monotone with respect to the sequence of its internal hypotheses: $h_i \geq h_{i-1}$ when considered as function on $\mathcal{X}_n$. The following theorem establishes the correctness and worst-case bound of the above prediction algorithm.

THEOREM 1. *Algorithm $\mathcal{P}$ learns the set of all monomials within the prediction model. It makes at most $n+1$ prediction errors.*

Proof. For the sake of completeness, we give a correctness proof here, which goes essentially back to Valiant [22].

Clearly, if the concept to be learned is "FALSE," no prediction error occurs and we are done. Otherwise, let $m = \ell_{i_1} \cdots \ell_{i_k}$, be the unique monomial describing the target concept c, where each ℓ_{i_j} is either x_{i_j} or $\bar{x}_{i_j}$. Since $\mathcal{P}$ computes all its internal hypotheses by possibly removing literals from h_0, it suffices to show that all the literals in the target monomial m survive and that every prediction error results in removing at least one literal from $\mathcal{P}$'s current internal hypothesis. This is done via the following claims.

Claim 1. *No literal in the target concept m is ever removed from h_0.*

Suppose the converse, i.e., there is a literal, say ℓ_i, in the target monomial m which at some stage is removed from h_0. Let ℓ_i be the first such literal. Consequently, before

ℓ_i is removed, all literals in m are still contained in $\mathcal{P}$'s internal hypothesis h. Thus, every example b satisfying $\mathcal{P}$'s internal hypothesis satisfies m, too, i.e., $h(b) = 1$ implies $m(b) = 1$. Thus, the removal must happen on an example b such that $m(b) = 1$, but $h(b) = 0$. However, while processing its loop $\mathcal{P}$ removes only those literals ℓ_j for which $\ell_j(b^j) = 0$. But for all literals ℓ_k in m we have $\ell_k(b^k) = 1$, and hence they survive. This contradiction proves Claim 1.

Claim 2. *Prediction errors are made on positive examples exclusively.*

By construction, $\mathcal{P}$ performs all its predictions by its internal hypotheses, which are based on monomials. The value of a monomial is 1 iff all its literals evaluate to 1. Hence, if $m(b) = 0$ at least one of its literals must evaluate to 0. By Claim 1, all literals contained in m are contained in all of $\mathcal{P}$'s internal hypotheses. Thus, at least one of the literals in $\mathcal{P}$'s actual internal hypothesis also evaluates to 0 causing $\mathcal{P}$ to predict 0. Hence, no prediction error can occur on a negative example.

Claim 3. *Each prediction error causes $\mathcal{P}$ to remove at least one literal from its actual internal hypothesis.*

Suppose the converse, i.e., after some prediction error no literal is removed. Thus, the example on which the prediction error occurred must satisfy all literals in $\mathcal{P}$'s actual internal hypothesis, since otherwise at least one literal is removed. Consequently, $\mathcal{P}$ has predicted 1, but since we had a prediction error, the true value of the target must be 0. Therefore, the prediction error had to occur on a negative example. This contradicts Claim 2.

Finally, we have to prove that at most $n + 1$ prediction errors can occur. Initially, $\mathcal{P}$'s internal hypothesis h_0 contains $2n$ literals. For each example exactly n literals evaluate to 1 and exactly n literals evaluate to 0. Hence, the first prediction error causes $\mathcal{P}$ to remove precisely n literals from its first internal hypothesis h_0. The worst-case obviously occurs if *all* literals must be eliminated from h_0, i.e., if m is the constant monomial 1 ($=$ TRUE). Now, the assertion directly follows by Claim 3. This proves the theorem. ∎

To learn the monotone concept class $\mathcal{MC}_n$, we use the following modified algorithm $\mathcal{MP}$ given a data sequence $d = \langle b_1, c(b_1), b_2, c(b_2), b_3, c(b_3), \dots \rangle$ for $c \in \mathcal{MC}_n$.

***Algorithm $\mathcal{MP}$*:**

Initialize $h_0 := x_1 \dots x_n$.
for $i = 1, 2, \dots$ **do**
  let h_{i-1} denote $\mathcal{MP}$'s internal hypothesis produced before receiving b_i;
  when receiving b_i predict $h_{i-1}(b_i)$;
  read $c(b_i)$;
  if $h_{i-1}(b_i) = c(b_i)$ **then** $h_i := h_{i-1}$
    else for $j := 1$ **to** n **do** **if** $b_i^j = 0$ **then** delete x_j in h_{i-1};
    let h_i be the resulting monomial
end.

Clearly, Theorem 1 can be directly reproved for $\mathcal{MC}$ with the only difference that now the worst-case bound for the number of prediction errors is n instead of $n + 1$.

5. Complexity Analysis: Best and Worst Case

For the learning models introduced above we want to estimate the best-case complexity, the worst-case complexity, and the expectation of algorithm $\mathcal{P}$ and $\mathcal{MP}$. Let us start considering the best and the worst case.

5.1. *On-line Prediction: Error Complexity*

$\mathcal{P}$ and $\mathcal{MP}$ do not make any prediciton errors iff the initial hypothesis h_0 equals the target monomial. For $\mathcal{P}$ this means that the concept to be learned is "FALSE", while for $\mathcal{MP}$ the concept is the all-1 vector. These special concepts can be considered as **minimal** in their corresponding class. For them the best-case and the worst-case number of predictions errors obviously do coincide since both algorithms are monotone and start with the minimal concept as their first internal hypothesis.

To study the general case, let us call the literals appearing in a monomial m **relevant**. All the other literals in $\mathcal{L}_n$ (resp. in $\{x_1, x_2, \ldots, x_n\}$ in the monotone case) will be called **irrelevant** for m. There are $2n - \#(m)$ irrelevant literals in the general case, and $n - \#(m)$ in the monotone setting. We call bit i **relevant** for m if x_i or $\bar{x}_i$ is relevant for m and use

$$\boldsymbol{k} \ := \ \boldsymbol{k(m)} \ := \ n - \#(m)$$

throughout the rest of the paper to denote the number of irrelevant bits.

Now we can expresses the best-case and worst-case number of prediction errors made by the Wholist algorithm as follows:

THEOREM 2. *Let $c = L(m)$ be a non-minimal concept in $\mathcal{MC}_n$. Then algorithm $\mathcal{MP}$ makes 1 prediction error in the best case, and $k(m)$ prediction errors in the worst-case. If c is a non-minimal concept of $\mathcal{C}_n$ algorithm $\mathcal{P}$ makes 2 prediction errors in the best case and $1 + k(m)$ prediction errors in the worst-case.*

Proof. For $\mathcal{MP}$ the worst-case occurs for a data sequences that forces the algorithm to delete irrelevant literals one at a time. Algorithm $\mathcal{P}$ gets rid of n irrelevant literals when it makes the first prediction error. Then $n - \#(m)$ irrelevant literals remains.

The best-case happens when the data sequences is such that by a single error $\mathcal{MP}$ can remove all irrelevant literals from h_0 (this requires that in the example all irrelevant variables get the value 0). For $\mathcal{P}$, it means that after its first prediction error, it can remove the remaining irrelevant literals in a single step. ∎

As the proof of Theorem 2 shows, the gap between the best-case and worst-case behavior can become quite large. Thus, it is natural to ask what are the expected bounds for the number of prediction errors on randomly generated data sequences. Before answering this question let us estimate the worst-case number of prediction errors averaged over the whole concept class $\mathcal{MC}_n$, resp. $\mathcal{C}_n$. Thus we get a complexity bound with respect to the problem parameter n, instead of the individual value $\#(m)$ as in Theorem 2. This averaging clearly depends on the underlying probability distribution for selecting the target concepts (note that for the corresponding data sequences we consider the worst input). The average will be shown to be linear in n if the literals are binomially distributed.

Basic knowledge with discrete probability theory will be assumed for the following analysis. Let us recall some analytic notions that will be used in several proofs. For a random variable X that takes natural numbers as its values it is often convenient to study its **probability generating function** (abbr. pgf) $\boldsymbol{G_X}$ defined as follows:

$$G_X(z) \ := \ \sum_{\ell \geq 0} \Pr[X = \ell] \ z^{\ell} \ . \tag{1}$$

Note that all the coefficients in (1) are nonnegative, and that they sum to 1, i.e., $G_X(1) = 1$. Thus, the power series (1) is absolutely convergent for all z with $|z| \leq 1$, where $|z|$ denotes the absolute value of z. Consequently, we may compute the first derivative of G_X by differentiating its summands, i.e.,

$$G'_X(z) \ = \ \sum_{\ell \geq 0} \Pr[X = \ell] \cdot \ell \cdot z^{\ell - 1} \ . \tag{2}$$

Moreover, $G'_X(z)$ is also absolutely convergent, and the radius of convergence of G_X and G'_X coincide. Thus, the expectation and variance of X can be computed as follows:

$$\begin{aligned} E[X] \ &= \ G'_X(1) \ , & (3) \\ V[X] \ &= \ G''_X(1) + G'_X(1) - G'_X(1)^2 \ , & (4) \end{aligned}$$

provided the power series obtained still do converge for $z = 1$. Furthermore, if X is any random variable that takes only nonnegative integer values, we can decompose its pgf into a sum of conditional pgf's with respect to any other discrete random variable Y as follows (cf. Graham, Knuth, Patashnik [7]):

$$G_X(z) \ = \ \sum_{y \in rg(Y)} \Pr[Y = y] \ g_{X|y}(z) \ . \tag{5}$$

Here $rg(Y)$ denotes the range of Y, and $g_{X|y}$ is the pgf for the random variable $X|y$, i.e., X under the knowledge that $Y = y$. Hence $g_{X|y}$ represents all the probabilities $\Pr[X = x \mid Y = y]$, $x \in rg(X)$. For further details concerning random variables and probability generating functions the reader is referred to [7].

To generate the probability distributions we assume for the class $\mathcal{MC}_n$ that the relevant positive literals are drawn independently at random with probability p for some $0 < p < 1$. Thus, with probability $q := 1 - p$ a positive literal is irrelevant. The length of the monomials drawn according to this distribution is binomially distributed with parameter p. Thus we will call such a distribution on the concept class a **binomial distribution**.

THEOREM 3. *Let the concepts in $\mathcal{MC}_n$ be binomially distributed with parameter p. Then the average number of prediction errors of $\boldsymbol{\mathcal{MP}}$ for the worst data sequences is $n(1-p)$.*

Proof. Let W_n be the random variable for the worst-case number of prediction errors that may occur. Thus, the range of W_n is $\{0, \ldots, n\}$. To compute $\Pr[W_n = k]$ we have to determine the probability to draw a target monomial having k irrelevant literals. This probability is clearly

$$\binom{n}{n-k} p^{n-k} (1-p)^k. \tag{6}$$

Thus, according to Theorem 2 the number of prediction errors is binomially distributed with parameter $1-p$. It is well known that in this case the expectation equals $n(1-p)$.

To illustrate the use of pgf, which we will need later in the more complicated cases, let us also perform an estimation of this expectation explicitely. One can write down the pgf for W_n as follows:

$$\begin{aligned} G_{W_n}(z) &= \sum_{k=0}^{n} \Pr[W_n = k] \cdot z^k = \sum_{k=0}^{n} \binom{n}{n-k} p^{n-k}(1-p)^k z^k \\ &= (p + (1-p)z)^n \end{aligned}$$

Now, it remains to compute $G'_{W_n}(1)$ which is $n \cdot (1-p)$. ▮

For the particular case $p = 1/2$, the bound says that the maximal number of prediction errors when uniformly averaged over all concepts in $\mathcal{MC}_n$ equals $n/2$.

Next, we turn our attention to the class $\mathcal{C}_n$. In order to compare it to the monotone case we have to clarify what it means for concepts in $\mathcal{C}_n$ to be binomially distributed. Since there are $3^n + 1$ many concepts in $\mathcal{C}_n$ for a uniform distribution each concept must have probability $1/(3^n+1)$. For each position $i = 1, \dots, n$ three options are possible, i.e., we may choose x_i, $\bar{x}_i$ or neither of them. This suggests the formula

$$\binom{n}{k_1, k_2, k_3} p_1^{k_1} p_2^{k_2} p_3^{k_3} , \tag{7}$$

where p_1 is the probability to take x_i, p_2 the probability to choose $\bar{x}_i$ and p_3 the probability to choose none, and $p_1 + p_2 + p_3 = 1$ and $k_1 + k_2 + k_3 = n$. $k_1 + k_2$ counts the number of relevant literals, resp. bits. However, this formula does not include the concept "FALSE." Thus, let us introduce $p_f \in (0,1)$ for the probability to choose "FALSE." Then, Formula (7) becomes

$$(1 - p_f)\binom{n}{k_1, k_2, k} p_1^{k_1} p_2^{k_2} p_3^{k} . \tag{8}$$

We call such a probability distribution a **weighted multinomial distribution** with parameters (p_f, p_1, p_2, p_3).

Theorem 4. *Let the concepts in $\mathcal{C}_n$ occur according to a weighted multinomial distribution with parameters (p_f, p_1, p_2, p_3). Then the average number of prediction errors of $\mathcal{P}$ for the worst data sequences is $(1-p_f)(1+np_3)$.*

Proof. Let W_n be the random variable for the worst-case number of prediction errors that may occur. Thus, the range of W_n is $\{0, \dots, n+1\}$. Since the first prediction error always causes the algorithm $\mathcal{P}$ to delete precisely n literals, the worst-case $k+1$ occurs iff there are k irrelevant literals in h_1. Thus, for computing $\Pr[W_n = k+1]$, it suffices to determine the probability of monomials in $\mathcal{C}_n$ that have $n-k$ relevant literals. Each such monomial not representing the concept "FALSE" must contain precisely $n-k$ pairwise different literals such that no literal is the negation of any other. Thus, Formula (8) applies, and hence, the pgf for W_n can be written as

$$G_{W_n}(z) = \sum_{k=0}^{n+1} \Pr[W_n = k] \cdot z^k = p_f + \sum_{k=1}^{n+1} \Pr[W_n = k] \cdot z^k$$

$$
\begin{aligned}
&= p_f + \sum_{k=0}^{n} \Pr[W_n = k+1] \cdot z^{k+1} \\
&= p_f + (1-p_f) \cdot \sum_{\substack{k_1+k_2+k=n \\ k_1,k_2,k \geq 0}} \binom{n}{k_1, k_2, k} p_1^{k_1} p_2^{k_2} p_3^{k} \cdot z^{k+1} \\
&= p_f + z \cdot (1-p_f)(p_1+p_2+p_3 z)^n
\end{aligned}
$$

Now, by Equation (3), we directly obtain $E[W_n]$ by computing $G'_{W_n}(1)$, i.e.,

$$E[W_n] \quad = \quad (1-p_f)(1+np_3) \ .$$

∎

For the particular case that all concepts from $\mathcal{C}_n$ are equally likely, i.e., $p_1 = p_2 = p_3 = 1/3$ and $p_f = 1/(3^n+1)$, we directly get that on the average

$$\frac{3^{n-1}}{3^n+1} \cdot (3+n) \ < \ \frac{n}{3} + 1$$

errors are to be expected given the worst data sequences. Hence, in this case the class $\mathcal{C}_n$ seems to be easier to learn than $\mathcal{MC}_n$ with respect to the complexity measure *prediction errors*. However, this impression is a bit misleading, since the probabilities to generate an irrelevant literal are *different*, i.e., $1/3$ for $\mathcal{C}_n$ and $1/2$ for $\mathcal{MC}_n$. If we assume the probabilities to generate an irrelevant literal to be equal, say q, and make the meaningful assumption that "FALSE" has the same probability than "TRUE" (that is, the *minimal* and the *maximal* concept in $\mathcal{C}_n$ are equally likely) then the average complexity is

$$\frac{1}{1+q^n}(1+nq) \quad \text{for } \mathcal{C}_n \qquad \text{and} \qquad nq \quad \text{for } \mathcal{MC}_n \,.$$

Note that for $\mathcal{C}_n$ it holds $q = 1-(p_1+p_2)$, and for $\mathcal{MC}_n$ $q = 1-p$. Thus, under these assumptions $\mathcal{MC}_n$ is easier to learn than $\mathcal{C}_n$. We think this insight is interesting, since it clearly shows the influence of the underlying distribution. In contrast, previous work has expressed these bounds in terms of the VC-dimension which is the same for both classes, i.e., n.

5.2. *Learning in the Limit: Mind Changes and Total Learning Time*

The results presented above directly translate to learning in the limit from informant for the complexity measure *number of mind changes*. This is also true for learning from positive presentations because only positive examples guide the algorithm and may cause an error/mind change.

What can be said about the total learning time? On the one hand, the best-case can be handled as above. That is, 1 *good* example is sufficient for learning $\mathcal{MC}_n$, and 2 for $\mathcal{C}_n$. For both algorithms $\boldsymbol{\mathcal{MP}}$ and $\boldsymbol{\mathcal{P}}$ the update time is linear in n. Thus, in the best case the total learning time is linear.

The worst-case total learning time is unbounded for both algorithms, since every positive presentation and every informant may contain as many repetitions of data that do not possess enough information to achieve the learning goal. Let us summarize:

THEOREM 5. *For every concept $c \in \mathcal{C}_n$ (resp. $c \in \mathcal{MC}_n$) it holds:*

(1) *There exists a positive presentation $d \in pos(c)$ such that $TT(\boldsymbol{\mathcal{P}}, d) = O(n)$ (resp. $TT(\boldsymbol{\mathcal{MP}}, d) = O(n)$).*

(2) *There exists an informant $d \in info(c)$ such that $TT(\boldsymbol{\mathcal{P}}, d) = O(n)$ (resp. $TT(\boldsymbol{\mathcal{MP}}, d) = O(n)$).*

(3) *For every $u \in \mathbb{N}$ there exists a positive presentation $d \in pos(c)$ such that $TT(\boldsymbol{\mathcal{P}}, d) > u$ (resp. $TT(\boldsymbol{\mathcal{MP}}, d) > u$), i.e., the worst-case total learning time is unbounded.*

(4) *For every $u \in \mathbb{N}$ there exists an informant $d \in info(c)$ such that $TT(\boldsymbol{\mathcal{P}}, d) > u$ (resp. $TT(\boldsymbol{\mathcal{MP}}, d) > u$).*

Hence, as far as learning in the limit and the complexity measure *total learning time* are concerned, there is a huge gap between the best-case and the worst-case behavior. Since the worst-case is unbounded, it does not make sense to ask for an analogue to Theorem 3 and 4. Instead, we continue by studying the average-case behavior of the limit learner $\boldsymbol{\mathcal{P}}$ and $\boldsymbol{\mathcal{MP}}$ and start with learning from positive presentations only.

6. Average-Case Analysis for Learning in the Limit from Positive Presentations

For the following average case analysis we assume that the data sequences are generated at random with respect to some probability distribution D taken from some class of admissible distributions $\mathcal{D}$, which will be specified later. We are interested in the *average number* of examples till an algorithm has converged to a correct hypothesis. Therefore, let **CONV** denote a random variable that counts the number of examples till convergence .

Let d be a positive presentation of the concept c to be learned that is generated at random according to D. If the concept to be learned is "FALSE" no examples are needed (and none exist). Otherwise, if the target concept contains precisely n literals then one positive example suffices (note that this one is unique). Thus, for these two cases everything is clear and the probability distributions D on the set of positive examples for c are trivial.

Thus it remains to analyse the nontrivial cases. Let $c = L(m) \in \mathcal{C}_n$ be a concept with monomial $m = \bigwedge_{j=1}^{\#(m)} \ell_{i_j}$. Let $k := k(m) = n - \#(m) > 0$. Note that there are 2^k positive examples for c. First, we consider the special distribution D where all positive examples are equally likely. That is, we study $\boldsymbol{\mathcal{P}}$'s behavior on data sequences $d = (b_j)_{j \in \mathbb{N}^+} \in pos(c)$ where the b_j are drawn independently and each b_j takes each of the possible 2^k values with probability 2^{-k}. We would like to compute the expected number of examples taken by $\boldsymbol{\mathcal{P}}$ until convergence.

The first example received forces $\boldsymbol{\mathcal{P}}$ to delete precisely n of the $2n$ literals in h_0. Thus, this example always plays a *special* role. Note that the resulting hypothesis h_1 depends on

b_1, but the number k of literals that remain to be deleted from h_1 until convergence is *independent* of b_1. It is therefore convenient to compute the expectation by considering $\mathcal{P}$'s behavior after having read the first example.

The next example will allow $\mathcal{P}$ to delete s irrelevant literals, where s can be any number between 0 and the total number k of remaining irrelevant literals. This observation suggests to look at the whole learning process as a Markov process. The states of the Markov process are defined by the number of irrelevant literals in $\mathcal{P}$'s internal hypothesis that remain to be deleted; let us call them $\sigma_0, \sigma_1, \ldots, \sigma_k$. We have to determine the transition probabilities $p(r, s)$ to go from state σ_r to state σ_s in one step. Clearly, $p(r, s) = 0$ for $r < s$.

LEMMA 1. *Let $c = L(m)$ be a nontrivial concept in $\mathcal{L}_n$. If all positive examples for m are equally likely then the transition probabilities $p(r, s)$ with $0 \le s \le r \le k$ are given by $p(r, s) = 2^{-r}\binom{r}{s}$.*

Proof. If the Markov process is in state r there are still r irrelevant literals to be deleted. Let $k' := k - r$ be the number of irrelevant literals already deleted before having entered state σ_r. All these literals can take any value without affecting $\mathcal{P}$'s internal hypothesis in state σ_r. If state σ_s has to be reached in one step the algorithm must delete $r - s$ many literals. Thus, there are $\binom{r}{r-s}$ many possibilities to choose the literals to be deleted and

$$p(r, s) \;=\; \frac{\binom{r}{r-s} \cdot 2^{k'}}{2^k} \;=\; \frac{\binom{r}{s} \cdot 2^{k-r}}{2^k} \;=\; \frac{1}{2^r} \cdot \binom{r}{s} \;.$$

∎

Let α_k be the expected number of steps to reach σ_0 starting in σ_k. Then, the following recursion formula holds:

$$\alpha_0 = 0 \qquad \text{and} \qquad \alpha_k \;=\; 1 + \sum_{\nu=0}^{k} p(k, \nu) \cdot \alpha_\nu \quad \text{for } k > 0 \;.$$

One can easily show an upper bound $\alpha_k \le 2(1 + \log_2 k)$. However, solving the recurrence exactly in closed form seems complicated. We can obtain a sufficiently good estimate on α_k by using *tail bounds.*

THEOREM 6. $\alpha_k \le \lceil \log_2 k \rceil + 3$ for all $k \in \mathbb{N}^+$.

Proof. Let us compute the probability that after having processed μ positive examples there is still at least one irrelevant literal in $\mathcal{P}$'s internal hypothesis. After the first example, there are k irrelevant literals left in h_1. On each positive example with probability $1/2$ an irrelevant literal evaluates to 0, and thus gets deleted. Consequently, for each remaining irrelevant literal the probability to survive $\mu + 1$ examples is $2^{-\mu}$, and the probability that at least one irrelevant literal survives all $\mu + 1$ examples can be bounded by $k\ 2^{-\mu}$. This bound is of interest only for $\mu > \log k$, otherwise we can use the trivial upper bound 1. Then, $E[\text{CONV} - 1]$ can be computed as

$$E[\text{CONV} - 1] \;=\; \sum_{\mu \ge 0} \Pr[\text{CONV} - 1 > \mu] \;.$$

Then we obtain

$$\begin{aligned}
E[\mathrm{CONV}-1] &= \sum_{\mu=0}^{\infty} \Pr[\mathrm{CONV}-1>\mu] \;\le\; \lambda + \sum_{\mu=\lambda}^{\infty} \Pr[\mathrm{CONV}-1>\mu] \\
&\le \lambda + k\cdot\sum_{\mu=\lambda}^{\infty} 2^{-\mu} \;=\; \lambda + \frac{k}{2^{\lambda-1}} \,.
\end{aligned}$$

Choosing $\lambda = \lceil \log_2 k \rceil$ yields $E[\mathrm{CONV}-1] \le \lceil \log_2 k \rceil + 2$. After adding the first example the bound claimed in the theorem follows. ∎

The generalization to larger classes of distributions is obvious. We shall exemplify it for the class of **binomial distributions**, where in a random positive examples all entries corresponding to irrelevant bits are selected independently of each other. With some probability p this will be a 1, and with probability $q := 1-p$ a 0. We will consider only nontrivial distributions where $0 < p < 1$. Note that otherwise the data sequence does not contain all positive examples.

Theorem 7. *Let $c = L(m)$ be a non-minimal concept in $\mathcal{C}_n$, and let the positive examples for c be binomially distributed with parameter p. Define $\psi := \min\{\frac{1}{1-p},\ \frac{1}{p}\}$. Then the expected number of positive examples needed by algorithm $\mathcal{P}$ until convergence can be bounded by*

$$E[\mathrm{CONV}] \;\le\; \lceil \log_\psi k(m) \rceil + 3 \,.$$

Proof. Let $k := k(m)$. The first positive example contains ν times a 1 and $k-\nu$ many 0 with probability $\binom{k}{\nu} p^\nu q^{k-\nu}$ at the positions not corresponding to a literal in the target monomial m. Now, assuming any such vector, we easily see that h_1 contains ν positive irrelevant literals and $k-\nu$ negative literals. Therefore, in order to achieve convergence, the algorithm $\mathcal{P}$ now needs positive examples that contain at least one 0 for each positive irrelevant literal and at least one 1 for each negative irrelevant literal. Thus, the probability that at least one irrelevant literal survives μ subsequent positive examples is bounded by $\nu q^\mu + (k-\nu)p^\mu$. Therefore,

$$\Pr[\mathrm{CONV}-1 \;>\mu] \;\le\; \sum_{\nu=0}^{k} \binom{k}{\nu} p^\nu q^{k-\nu} \cdot (\nu q^\mu + (k-\nu)p^\mu) \,.$$

Next, we derive a closed formula for the sum given above.

Claim 1. $\displaystyle\sum_{\nu=0}^{k} \binom{k}{\nu} p^\nu q^{k-\nu}\cdot \nu = kp$ and $\displaystyle\sum_{\nu=0}^{k} \binom{k}{\nu} p^\nu q^{k-\nu}\cdot (k-\nu) = kq$

The first equality can be shown as follows.

$$\begin{aligned}
\sum_{\nu=0}^{k} \binom{k}{\nu} p^\nu q^{k-\nu}\cdot \nu &= \sum_{\nu=1}^{k} \binom{k}{\nu} p^\nu q^{k-\nu}\cdot \nu = \sum_{\nu=0}^{k-1} \binom{k}{\nu+1} p^{\nu+1} q^{k-1-\nu}\cdot (\nu+1) \\
&= \sum_{\nu=0}^{k-1} k\cdot \binom{k-1}{\nu} p^{\nu+1} q^{k-(\nu+1)} \;=\; k\,p\cdot \sum_{\nu=0}^{k-1} k\cdot \binom{k-1}{\nu} p^{\nu} q^{(k-1)-\nu} \\
&= k\,p\cdot (p+q)^{k-1} \;=\; k\,p \,.
\end{aligned}$$

The other equality can be proved analogously, which yields Claim 1.

Now, proceeding as above, we obtain

$$\begin{aligned}
E[\mathrm{CONV}-1\,] &= \sum_{\mu=0}^{\infty} \Pr[\mathrm{CONV}-1\ > \mu] \ \le\ \lambda + \sum_{\mu=\lambda}^{\infty}\sum_{\nu=0}^{k}\binom{k}{\nu} p^{\nu} q^{k-\nu}\cdot(\nu q^{\mu} + (k-\nu)p^{\mu}) \\
&= \lambda + \sum_{\mu=\lambda}^{\infty}\sum_{\nu=0}^{k}\binom{k}{\nu} p^{\nu} q^{k-\nu}\cdot \nu q^{\mu} + \sum_{\mu=\lambda}^{\infty}\sum_{\nu=0}^{k}\binom{k}{\nu} p^{\nu} q^{k-\nu}\cdot (k-\nu) p^{\mu} \\
&= \lambda + \sum_{\mu=\lambda}^{\infty} q^{\mu}\cdot \underbrace{\sum_{\nu=0}^{k}\binom{k}{\nu} p^{\nu} q^{k-\nu}\cdot \nu}_{=kp \quad \text{by Claim 1}} + \sum_{\mu=\lambda}^{\infty} p^{\mu}\cdot \underbrace{\sum_{\nu=0}^{k}\binom{k}{\nu} p^{\nu} q^{k-\nu}\cdot (k-\nu)}_{=kq \quad \text{by Claim 1}} \\
&= \lambda + kp\cdot\sum_{\mu=\lambda}^{\infty} q^{\mu} + kq\cdot\sum_{\mu=\lambda}^{\infty} p^{\mu} \ =\ \lambda + k\cdot[q^{\lambda}+p^{\lambda}] \ \le\ \lambda + k\,2\,\psi^{-\lambda}\,.
\end{aligned}$$

Finally, choosing $\lambda = \lceil \log_{\psi} k \rceil$ gives the statement of the theorem. ∎

Note that the bound of Theorem 7 coincides with the bound given in Theorem 6 for the special case $p = 1/2$. A similar analysis can be given in the monotone setting for algorithm $\mathcal{MP}$. As a corollary we get

COROLLARY 8. *For every binomially distributed positive presentation with parameter $0 < p < 1$ the average total learning time of algorithm $\mathcal{P}$ for concepts in $\mathcal{C}_n$ is at most $O(n \log n)$. More precisely, a concept $c = L(m)$ requires time $O(n \log(n - \#(m) + 2))$ on the average.*

The expectation alone does not provide complete information about the average case behavior of an algorithm. It is helpful to know larger moments, too, in particular the variance. Then one can deduce bounds how often the algorithm exceeds the average considerably by applying, for example, Chebyshev's inequality. The Wholist algorithm possesses two favorable properties that simplify the analysis considerably, it is **set-driven** and **conservative**. They allow to establish good bounds for the tail probabilities.

Set-driven means that the output depends only on the *range* of the input sequence. More formally, for all $c \in \mathcal{C}_n$ all $d, h \in pos(c)$ and all $i, j \in \mathbb{N}^+$ the equality $d[i]^+ = h[j]^+$ of the range of the two prefices implies $\mathcal{P}(d[i]) = \mathcal{P}(h[j])$. It is easy to see that the Wholist algorithm fulfills this property.

Furthermore, a learner is said to be conservative if every mind change is caused by an inconsistency with the data seen so far. The Wholist algorithm satisfies this condition, too, i.e., for all $c \in \mathcal{C}_n$, all $d \in pos(c)$ and all $i,\, j \in \mathbb{N}^+$ it holds: if $\mathcal{P}(d[i]) \neq \mathcal{P}(d[i+j])$ then $d[i+j]^+ \not\subseteq L(\mathcal{P}(d[i]))$.

Now, we can apply the following theorem to establish exponentially shrinking tail bounds for the expected number of examples needed in order to achieve convergence.

THEOREM 9. (Rossmanith and Zeugmann [19])

Let CONV *be the sample complexity of a conservative and set-driven learning algorithm. Then for arbitrary $t \in \mathbb{N}$ it holds*

$$\Pr[\mathrm{CONV} > 2\,t\cdot E[\mathrm{CONV}]\,] \ \le\ 2^{-t}\,.$$

A simple calculation shows that in case of exponentially shrinking tail bounds the variance is bounded by $O(E[\mathrm{CONV}]^2)$.

7. Stochastic Finite Learning

One can also convert the Wholist algorithm into a learner that from positive data identifies all concepts in $\mathcal{C}_n$ *stochastically in a bounded number of rounds with high confidence.* A certain amount of *additional knowledge* concerning the underlying class of probability distributions is required. Thus, in contrast to the PAC model, the resulting learning model is *not* distribution-free. On the other hand, it is stronger than the PAC model by requiring the output to be *probably exactly correct* rather than *probably approximately correct.* But the main advantage is the usage of the additional knowledge to reduce the sample size, and hence the total learning time drastically. This nicely contrasts to previous work undertaken in the area of PAC learning (cf., e.g., [2, 5, 9, 12, 14, 20, 21]. These papers have shown certain concepts classes to be PAC learnable from a polynomial number of examples given a known distribution or class of distributions, while the general PAC learnability of these concepts classes cannot be achieved or remains open. As a matter of fact, our general approach, i.e., performing a thorough average-case analysis and proving exponentially shrinking tail bounds for the expected total learning time, can be applied to obtain results along this line, too. That is, we can design algorithms learning pattern languages and subsets thereof stochastically with high confidence in a constant number of rounds (cf. [18, 19]).

Definition 3. *Let $\mathcal{D}$ be a set of probability distributions on the learning domain, $\mathcal{C}$ a concept class, $\mathcal{H}$ a hypothesis space for $\mathcal{C}$, and $\delta \in (0,1)$. $(\mathcal{C},\mathcal{D})$ is said to be* ***stochastically finite learnable with δ-confidence*** *with respect to $\mathcal{H}$ iff there is an IIM M that for every $c \in \mathcal{C}$ and every $D \in \mathcal{D}$ performs as follows. Given a random presentation d for c generated according to D, M stops after having seen a finite number of examples and outputs a single hypothesis $h \in \mathcal{H}$. With probability at least $1-\delta$ (with respect to distribution D) h has to be correct, that is $L(h) = c$ in case of monomials.*

If stochastic finite learning can be achieved with δ-confidence for every $\delta > 0$ then we say that $(\mathcal{C},\mathcal{D})$ can be learned stochastically finite **with high confidence**.

Again, we will study the case that the positive examples are binomially distributed with parameter p. However, we do not require precise knowledge about the underlying distribution. But it is reasonable to assume that at least some information is available. Let *prior knowledge* be provided by two parameters p_{low} and p_{up} such that the true parameter p satisfies $p_{low} \le p \le p_{up}$. Binomial distributions fulfilling this requirement will be called $(\boldsymbol{p_{low}, p_{up}})$**–admissible distributions**. Let $\boldsymbol{\mathcal{D}_n[p_{low}, p_{up}]}$ denote the set of such distributions on the learning domain $\mathcal{X}_n$.

If probability bounds p_{low} and p_{up} are available the Wholist algorithm can be transformed into a stochastical finite learner that identifies all concepts with high confidence.

Theorem 10. *Let $0 < p_{low} \le p_{up} < 1$ and define $\psi := \min\{\frac{1}{1-p_{low}}, \frac{1}{p_{up}}\}$. Then $(\mathcal{C}_n, \mathcal{D}_n[p_{low}, p_{up}])$ is stochastically finitely learnable with high confidence from positive presentations using $O(\log_2 1/\delta \cdot \log_\psi n)$ many examples.*

Proof. The learner is based on the Wholist algorithm and a counter for the number of examples already processed. If the Wholist algorithm is run for $\tau := \lceil \log_\psi n \rceil + 3$ many examples Theorem 7 implies that τ is at least as large as the expected convergence stage $E[\text{CONV}]$.

In order to achieve the desired confidence, the learner sets $\gamma := \lceil \log_2 \frac{1}{\delta} \rceil$ and runs the Wholist algorithm for a total of $2\,\gamma \cdot \tau$ examples. The algorithm outputs the last hypothesis $h_{2\,\gamma\cdot\tau}$ produced by the Wholist algorithm and stops thereafter. The reliability follows from the tail bounds established in Theorem 9. ∎

In practice, often one is given a fixed set of examples from which a best guess has to be computed. Thus, we would also like to have a converse estimation that, based on the number l of examples, provides a bound on the error probability of the hypothesis computed. Let us denote this quantity by $error(l)$. From the analysis of the Wholist algorithm above it follows that the error is monotonically decreasing with l. If one stops after the l-th example an error occurs iff the algorithm has not converged yet, that is $\text{CONV} > l$. Thus,

$$error(l) \;=\; \Pr[\text{CONV} > l]\ .$$

For an irrelevant bit i the probability that not both x_i and $\bar{x}_i$ are eliminated within l steps equals $p^l + q^l$. If the monomial has k irrelevant bits the probability that exactly r irrelevant bits survive and hence the l-th hypothesis has r additional literals in comparison to the correct hypothesis is given by

$$\binom{k}{r}\;(p^l + q^l)^{\,r}\ .$$

By the inclusion exclusion principle,

$$error(l) \;=\; \sum_{r=1}^{k} (-1)^{r+1} \binom{k}{r}\;(p^l + q^l)^{\,r}\ .$$

Taking only the first two terms, we get upper and lower bounds

$$k\,(p^l + q^l) \;-\; \binom{k}{2}\;(p^l + q^l)^{\,2} \;\le\; error(l) \;\le\; k\,(p^l + q^l)\ .$$

Let $\bar{p} := \max\{p, q\}$ and $l = \lceil \log_{1/\bar{p}} k \rceil + a$ with $a \in \mathbb{N}$. Then

$$error(l) \;\le\; 2\,k\,\bar{p}^{\,l} \;\le\; 2\,\bar{p}^{\,a}\ .$$

Thus, if the number of examples exceeds the (upper bound on the) expectation (Theorem 7) the error probability drops down exponentially.

Calculating the lower bound shows that the upper bound estimation is close to best possible.

$$error(l) \;\ge\; k\,\bar{p}^{\,l} -\; k^{\,2}\,\bar{p}^{\,2l} \;\ge\; \bar{p}^{\,a} - \bar{p}^{\,2a}\ .$$

Thus we get

THEOREM 11. *For every concept $c = L(m)$, a hypothesis computed by the Wholist algorithm after having processed $l = \lceil \log_{1/\bar{p}} k(m) \rceil + a$ examples satisfies the error bound*

$$\bar{p}^{\,a} - \bar{p}^{\,2a} \;\le\; error(l) \;\le\; \bar{p}^{\,a}\ .$$

In particular, for all $j \leq \lceil \log_{1/\bar{p}} k(m) \rceil + 1$ *it holds:* $\quad error(j) \geq \bar{p} - \bar{p}^2$.

In general, the learner does not know the number of irrelevant bits $k(m)$ of the monomial m he has to learn. Bounding $k(m)$ by the trivial value n gives

Corollary 12. *If the Wholist algorithm outputs a hypothesis after having seen* $l = \lceil \log_{1/\bar{p}} n \rceil + a$ *examples the error probability is guaranteed to be not larger than* $\bar{p}^a$.

The lower bound on the error probability implies a lower bound for the expectation for the round of convergence as follows:

$$E[\mathrm{CONV}] \;=\; \sum_{l=0}^{\infty} error(l) \;\geq\; (\bar{p} - \bar{p}^2)\, \lceil \log_{1/\bar{p}} k(m) \rceil .$$

Thus, the upper bound $\log_{1/\bar{p}} k(m) + O(1)$ in Theorem 7 is asymptotically tight.

The upper bound on the error probability also provides a direct and better estimate for stochastic finite learning for learning monomials (Theorem 10).

Corollary 13. *Let* $0 < p_{low} \leq p_{up} < 1$, $\psi := \min\{\frac{1}{1-p_{low}}, \frac{1}{p_{up}}\}$, *and* $\delta > 0$. *Then* $(\mathcal{C}_n, \mathcal{D}_n[p_{low}, p_{up}])$ *is stochastically finite learnable from positive presentations with* δ *-confidence using* $\log_\psi n + \log_\psi 1/\delta + O(1)$ *many examples.*

Thus, for learning monomials the confidence requirement increases the sample size by an additive term $\log_\psi 1/\delta$ only, instead of a multiplicative one $\log_2 1/\delta$ in general.

8. Average-case Analysis for Learning in the Limit from Informant

Finally, let us consider how the results obtained so far translate to the case of learning from informant. Since the Wholist algorithm does not learn anything from negative examples, one may expect that it behaves much poorer in this setting. Let us first investigate the uniform distribution over $\mathcal{X}_n$. Again, we have the trivial cases that the target concept is "FALSE" or m is a monomial without irrelevant bits. In the first case, no example is needed at all, while in the latter one, there is only one positive example having probability 2^{-n} . Thus the expected number of examples needed until successful learning is $2^n = 2^{\#(m)}$.

Theorem 14. *Let* $c = L(m) \in \mathcal{C}_n$ *be a nontrivial concept. If a data sequence for* c *is generated from the uniform distribution on the learning domain by independent draws the expected number of examples needed by algorithm* $\mathcal{P}$ *until convergence is bounded by*

$$E[\mathrm{CONV}] \;\leq\; 2^{\#(m)}\; (\lceil \log_2 k(m) \rceil + 3) \ .$$

Proof. Let CONV+ be a random variable for the number of positive examples needed until convergence. Every positive example is preceded by a possibly empty block of negative examples. Thus, we can partition the initial segment of any randomly drawn informant read until convergence into CONV+ many blocks B_j containing a certain number of negative examples followed by precisely one positive example. Let Λ_j be a random variable for the length of block B_j . Then $\mathrm{CONV} = \Lambda_1 + \Lambda_2 + \cdots + \Lambda_{\mathrm{CONV+}}$, where the Λ_j are independently identically distributed. In order to compute the distribution of Λ_j , it suffices to calculate the probabilities to draw a negative and a positive example, respectively. Since the overall number of positive examples for c is 2^k with $k = k(m)$, the probability to generate a positive example is 2^{k-n}. Hence, the probability to draw a negative example is $1 - 2^{k-n}$. Consequently,

$$\Pr[\Lambda_j = \mu + 1] \;=\; \left(1 - 2^{k-n}\right)^{\mu} \cdot 2^{k-n}\ .$$

Therefore,

$$\begin{aligned}
E[\text{CONV}] \;&=\; E[\Lambda_1 + \Lambda_2 + \cdots + \Lambda_{\text{CONV+}}] \\
&=\; \sum_{\zeta=0}^{\infty} E[\Lambda_1 + \Lambda_2 + \cdots + \Lambda_\zeta \mid \text{CONV+} = \zeta] \;\cdot\; \Pr[\text{CONV+} = \zeta] \\
&=\; \sum_{\zeta=0}^{\infty} \zeta \;\cdot\; E[\Lambda_1] \;\cdot\; \Pr[\text{CONV+} = \zeta] \\
&=\; E[\text{CONV+}] \;\cdot\; E[\Lambda_1]
\end{aligned}$$

By Theorem 6, we have $E[\text{CONV+}] \le \lceil \log_2 k \rceil + 3$, and thus it remains to estimate $E[\Lambda_1]$. A simple calculation shows

LEMMA 2. For every $0 < a < 1$ it holds:

$$\sum_{\mu=0}^{\infty} (\mu + 1) \cdot a^{\mu} \;=\; (1 - a)^{-2}\ .$$

Using this estimation we can conclude

$$\begin{aligned}
E[\Lambda_1)] \;&=\; \sum_{\mu=0}^{\infty} (\mu + 1) \;\cdot\; \Pr[\Lambda_1 = \mu + 1] \\
&=\; 2^{k-n} \sum_{\mu=0}^{\infty} (\mu + 1) \cdot \left(1 - 2^{k-n}\right)^{\mu} \;=\; 2^{n-k}\ ,
\end{aligned}$$

and thus the theorem follows. ∎

Hence, as long as the length of m is constant, and therefore $k(m) = n - O(1)$, we still achieve an expected total learning time of order $n \log n$. But if $\#(m)$ grows linearly the expected total learning becomes exponential. On the other hand, if there are many relevant literals then even h_0 may be considered as a not too bad *approximation* for c. Consequently, it is natural at this point to introduce an error parameter $\varepsilon \in (0, 1)$ as in the PAC model, and to ask whether one can achieve an expected sample complexity for computing an ε-approximation that is bounded by a function depending on $\log n$ and $1/\varepsilon$.

To answer this question, let us formally define $error_m(h_j) = D(L(h_j) \bigtriangleup L(m))$ to be the error made by hypothesis h_j with respect to monomial m. Here $L(h_j) \bigtriangleup L(m)$ stands for the symmetric difference of $L(h_j)$ and $L(m)$ and D for the underlying probability distribution with respect to which the examples are drawn. Note that by Theorem 1 we can conclude $error_m(h_j) = D(L(m) \setminus L(h_j))$. We call h_j an **ε-approximation** for m if $error_m(h_j) \le \varepsilon$. Finally, we redefine the stage of convergence. Let $d = (d_j)_{j \in \mathbb{N}^+}$ be an informant for $L(m)$, then

$$\mathbf{CONV}_{\boldsymbol{\varepsilon}}\boldsymbol{(d)} =_{df} \text{the least number } j \text{ such that} \quad error_m(P(d[i])) \le \varepsilon \text{ for all } i \ge j\ .$$

Note that once the Wholist algorithm has reached an ε-approximate hypothesis all further hypotheses will also be at least that close to the target monomial. The following theorem gives an affirmative answer to the question posed above.

THEOREM 15. *Let $c = L(m) \in \mathcal{C}_n$ be a nontrivial concept. Assuming that examples are drawn independently from the uniform distribution, the expected number of examples needed by algorithm $\boldsymbol{\mathcal{P}}$ until converging to an ε-approximation for c can be bounded by*

$$E[\mathrm{CONV}_\varepsilon] \;\leq\; \frac{1}{\varepsilon}\cdot(\lceil\log_2 k(m)\rceil + 3)\ .$$

Proof. It holds $error_m(h_0) = 2^{k(m)-n}$, since h_0 misclassifies exactly the positive examples. Therefore, if $error_m(h_0) \leq \varepsilon$, we are already done. Now suppose $error_m(h_0) > \varepsilon$. Consequently, $1/\varepsilon > 2^{n-k(m)}$, and thus the bound stated in the theorem is larger than $2^{n-k(m)}(\lceil\log_2 k(m)\rceil + 3)$, which, by Theorem 14 is the expected number of examples needed until convergence to a correct hypothesis. ∎

Thus, additional knowledge concerning the underlying probability distribution pays off again. Applying Theorem 9 and modifying Section 7 *mutatis mutandis*, we get a learner identifying ε-approximations for all concepts in $\mathcal{C}_n$ stochastically with high confidence using $O(\frac{1}{\varepsilon}\cdot\log\frac{1}{\delta}\cdot\log n)$ many examples. Comparing this bound with the sample complexity given in the PAC model, we see that it is reduced exponentially, i.e., instead of a factor n now we have the factor $\log n$.

Finally, we can generalize the last results to the case that the data sequences are binomially distributed for some parameter $p \in (0,1)$. This means that any particular vector containing ν times a 1 and $n-\nu$ a 0 has probability $p^\nu(1-p)^{n-\nu}$ since a 1 is drawn with probability p and a 0 with probability $1-p$. First, Theorem 14 generalizes as follows.

THEOREM 16. *Let $c = L(m) \in \mathcal{C}_n$ be a nontrivial concept. Let m contain precisely π positive literals and τ negative literals. If the labeled examples for c are independently binomially distributed with parameter p and $\psi := \min\{\frac{1}{1-p},\ \frac{1}{p}\}$ then the expected number of examples needed by algorithm $\boldsymbol{\mathcal{P}}$ until convergence can be bounded by*

$$E[\mathrm{CONV}] \;\leq\; \frac{1}{p^\pi(1-p)^\tau}\,\Big(\lceil\log_\psi k(m)\rceil + 3\Big)\ .$$

Proof. Assuming the same notation as in the proof of Theorem 14, it is easy to see that we only have to recompute $E[\Lambda_1]$, and thus $\Pr[\Lambda_1 = \mu+1]$, too. Since m contains precisely π positive literals and τ negative literals, the probability to draw a positive example is clearly $p^\pi(1-p)^\tau$, and thus the probability to randomly draw a negative example is $1-p^\pi(1-p)^\tau$. Consequently,

$$\Pr[\Lambda_1 = \mu+1] \;=\; (1-p^\pi(1-p)^\tau)^\mu\cdot p^\pi(1-p)^\tau\ ,$$

and Lemma 2 gives $E[\Lambda_1] = \frac{1}{p^\pi(1-p)^\tau}$. ∎

Theorem 15 directly translates into the setting of binomially distributed inputs, too.

THEOREM 17. *Let $c = L(m) \in \mathcal{C}_n$ be a nontrivial concept. Assume that the examples are drawn with respect to a binomial distribution with parameter p, and let $\psi = \min\{\frac{1}{1-p},\ \frac{1}{p}\}$.*

Then the expected number of examples needed by algorithm $\mathcal{P}$ *until converging to an* ε*-approximation for* c *can be bounded by*

$$E[\mathrm{CONV}] \ \leq \ \frac{1}{\varepsilon} \cdot (\lceil \log_\psi k(m) \rceil + 3) \ .$$

Finally, one can also get stochastic finite approximations with high confidence from informant with an exponentially smaller sample complexity.

THEOREM 18. *Let* $0 < p_{low} \le p_{up} < 1$ *and* $\psi := \min\{\frac{1}{1-p_{low}}, \ \frac{1}{p_{up}}\}$. *For* $(\mathcal{C}_n, \mathcal{D}_n[p_{low}, p_{up}])$ ε*-approximations are stochastically finitely learnable with* δ*-confidence from informant for any* ε, $\delta \in (0,1)$. *For this purpose,*

$$O\left(\frac{1}{\varepsilon} \cdot \log_2 1/\delta \cdot \log_\psi n\right) \ , \quad \textit{resp.} \quad O\left(\frac{1}{\varepsilon} \cdot (\log_\psi 1/\delta + \log_\psi n)\right)$$

many examples suffice.

9. Conclusions

A new framework for analyzing the average-case behavior of the well-known Wholist algorithm has been introduced. Based on it, we have studied the problem of learning concepts describable by monomials from positive presentations only, from positive and negative data within the setting of learning in the limit and on-line prediction. The complexity measures considered comprise the number of prediction errors, the number of mind changes as well as the total learning time. Tight bounds have been obtained showing that worst-case bounds differ exponentially from the average-case behavior in the particular setting of learning under binomially distributed input data.

Within our framework, we could completely overcome the drawbacks of the approach undertaken by Pazzani and Sarret [17]. In particular, our analysis is fully analytical and does not involve any computer simulation. The results obtained provide a general overview about the average-case behavior of the Wholist algorithm for the class of binomial distributions. The influence of the parameter p determining the particular binomial distribution on hand to the sample complexity has been elaborated (cf. Theorems 7 and 16).

Moreover, our approach has established *exponentially* shrinking tail bounds for the expected number of examples needed until convergence for both settings learning from positive data and from informant. These tail bounds in a natural way translate into stochastic finite learning in case of inference from text and PAC learning in case of informant. In case of binomial distributions with additional knowledge given by lower and upper bounds for the probability to set a variable to value 1, nicely reduces the sample complexity for stochastic finite inference and PAC learning to an amount that is exponentially smaller than previously obtained bounds (cf. Theorems 10 and 18). Clearly, these results directly translate into the monotone setting, i.e., to the class $\mathcal{MC}_n$.

Furthermore, by duality, our results imply the same exponential improvement for learning concepts over $\mathcal{X}_n$ that are describable by a disjunction of literals from $\mathcal{L}_n$. If the concept class is restricted to monotone disjunctions m the smallest worst-case bound for the number of prediction errors is achieved by Littlestone's Winnow algorithm [13], which is

$O(\#(m)\ \log n)$ ($\#(m)$ denoting the number of literals in the target disjunction), while our methods achieve $O(\log(n - \#(m)))$ many prediction errors on average. Hence, if the length of the target is a constant, Winnow gives roughly the same bound as our learner. Note that $O(\#(m)\ \log n)$ is a worst-case bound, but on the other hand Winnow cannot gain much on average. Even in the most favourable case of $\#(m)$ being a constant and a uniform distribution on the input data it will make $\Theta(\log n)$ many prediction errors on average.

Finally, it should be noted that our general technique to obtain very efficient stochastic finite learning, that is performing a thorough average-case analysis and proving exponentially shrinking tail bounds for the expected sample complexity, has a large range of potential applications. As far as Boolean concepts are concerned, promising candidates are k-CNF, k-DNF, k-Decison lists as well as k-term DNF, and k-term CNF.

Last but not least, one can also perform experiments using an implementation of the Wholist algorithm by visiting our WEB-page:

http://www.i.kyushu-u.ac.jp/ thomas/BOOLE/menue2.html

to convince oneself that the upper bounds obtained in this paper predict the actual behaviour of the Wholist algorithm quite well.

References

[1] J.M. Barzdin and R.V. Freivald, On the prediction of general recursive functions. *Soviet Math. Doklady* 13:1224-1228, 1972.

[2] G. Benedek and A. Itai. Learnability by fixed distributions. *in* "Proc. of the 1988 Workshop on Computational Learning Theory," pp. 81–90, Morgan Kaufmann 1988.

[3] J. Case, S. Jain, S. Lange and T. Zeugmann. Incremental concept learning for bounded data mining. to appear in Information and Computation.

[4] R. Daley and C.H. Smith. On the complexity of inductive inference. *Information and Control*, 69:12–40, 1986.

[5] F. Denis and R. Gilleron. PAC learning under helpful distributions. *in* "Proc. 8th International Workshop on Algorithmic Learning Theory - ALT'97," (M. Li and A. Maruoka, Eds.), Lecture Notes in Artificial Intelligence 1316, pp. 132–145, Springer-Verlag 1997.

[6] E. M. Gold, Language identification in the limit. *Information and Control* 10:447–474, 1967.

[7] R.L. Graham, D.E. Knuth and O. Patashnik. *Concrete Mathematics*, Addison-Wesley, Reading, Massachusetts, 1989.

[8] D. Haussler. Bias, version spaces and Valiant's learning framework. *in* "Proc. 8th National Conference on Artificial Intelligence, pp. 564–569, Morgan Kaufmann, 1987.

[9] M. Kearns, M. Li, L. Pitt and L.G. Valiant. On the learnability of Boolean formula. *in* "Proc. of the 19th Annual ACM Symposium on the Theory of Computing," pp. 285–295, ACM Press 1987.

[10] S. Lange and T. Zeugmann. Incremental learning from positive data. *Journal of Computer and System Sciences* 53(1):88–103, 1996.

[11] S. Lange and T. Zeugmann. Set-driven and rearrangement-independent learning of recursive languages. *Mathematical Systems Theory* 29(6):599–634, 1996.

[12] M. Li and P. Vitanyi. Learning simple concepts under simple distributions. *SIAM Journal on Computing*, 20(5):911-935, 1991.

[13] N. Littlestone. Learning quickly when irrelevant attributes are abound: A new linear threshold algorithm. *Machine Learning* 2:285–318, 1988.

[14] B. Natarajan. On learning Boolean formula. *in* "Proc. of the 19th Annual ACM Symposium on the Theory of Computing," pp. 295–304, ACM Press 1987.

[15] S. Okamoto and K. Satoh. An average-case analysis of k-nearest neighbor classifier. *in* "Proc. 1st International Conference on Case-Based Reasoning Research and Development," Lecture Notes in Computer Science 1010, pp. 253–264, Berlin, Springer-Verlag 1995.

[16] S. Okamoto and N. Yugami. Theoretical analysis of the nearest neighbor classifier in noisy domains. *in* "Proc. 13th International Conference on Machine Learning, pp. 355–363, Morgan Kaufmann 1996.

[17] M.J. Pazzani and W. Sarrett, A framework for average case analysis of conjunctive learning algorithms. *Machine Learning* 9:349–372, 1992.

[18] R. Reischuk and T. Zeugmann. Learning one-variable pattern languages in linear average time. *in* "Proc. 11th Annual Conference on Computational Learning Theory - COLT'98," pp. 198–208, ACM Press 1998.

[19] P. Rossmanith and T. Zeugmann. Learning k-variable pattern languages efficiently stochastically finite on average from positive data. *in* "Proc. 4th International Colloquium on Grammatical Inference - ICGI'98," (V. Honavar and G. Slutzki, Eds.), Lecture Notes in Artificial Intelligence 1433, pp. 13–24, Springer-Verlag 1998.

[20] Y. Sakai and A. Maruoka. Learning monotone log-term DNF formulas. *in* "Proc. of the 7th Annual ACM Conference on Computational Learning Theory," pp. 165–172, ACM Press 1994.

[21] Y. Sakai, E. Takimoto and A. Maruoka. Proper learning algorithms for functions of k terms under smooth distributions. *in* "Proc. of the 8th Annual ACM Conference on Computational Learning Theory," pp. 206–213, ACM Press 1995.

[22] L.G. Valiant. A theory of the learnable. *Communications of the Association of Computing Machinery* 27:1134-1142, 1984.