Theoretical Computer Science 411 (2010) 2384–2405

Contents lists available at ScienceDirect

Theoretical Computer Science

journal homepage: www.elsevier.com/locate/tcs

Playing monotone games to understand learning behaviors

Bruno Apolloni [a,*], Simone Bassis [a], Sabrina Gaito [a], Dario Malchiodi [a], Italo Zoppis [b]

[a] Dip. di Scienze dell'Informazione, Università degli Studi di Milano, Via Comelico 39/41 20135 Milano, Italy

[b] Dip. di Informatica, Sistemistica e Comunicazione, Università degli Studi Milano Bicocca. Piazza dell'Ateneo Nuovo 1 20126 Milano, Italy

ARTICLE INFO

Article history:
Received 27 December 2007
Accepted 12 February 2010
Communicated by T. Baeck

Keywords:
Overtraining control
Training stopping rule
Monotone games
Algorithmic inference
Computational learning
Subsymbolic learning

ABSTRACT

We deal with a special class of games against nature which correspond to subsymbolic learning problems where we know a local descent direction in the error landscape but not the amount gained at each step of the learning procedure. Namely, Alice and Bob play a game where the probability of victory grows monotonically by unknown amounts with the resources each employs. For a fixed effort on Alice's part Bob increases his resources on the basis of the results of the individual contests (victory, tie or defeat). Quite unlike the usual ones in game theory, his aim is to stop as soon as the defeat probability goes under a given threshold with high confidence. We adopt such a game policy as an archetypal remedy to the general overtraining threat of learning algorithms. Namely, we deal with the original game in a computational learning framework analogous to the Probably Approximately Correct formulation. Therein, a wise use of a special inferential mechanism (known as *twisting argument*) highlights relevant statistics for managing different trade-offs between observability and controllability of the defeat probability. With similar statistics we discuss an analogous trade-off at the basis of the stopping criterion of subsymbolic learning procedures. As a conclusion, we propose a principled stopping rule based solely on the behavior of the training session, hence without distracting examples into a test set.

1. Introduction

When we run learning algorithms on subsymbolic systems such as neural networks, usually we are not able to strictly identify the effects of given free parameter changes on the goal function we want to minimize. Rather, we incrementally optimize a statistic about this function taken on the training set (e.g. the sum of square errors) expecting that a companion optimization occurs in the goal function (e.g. the same function averaged on the input/output probability distribution). If the statistic gets to a plateau we wonder if it is the best we may obtain; then we must either stop or keep expecting to further improve the goal function. Continuing the training in the second case may indeed generate both *overfitting* phenomena and obvious waste of computational time. We study the general task of learning in these conditions through a special learning problem constituted by a match between two people who must regulate the prospective efforts in the single contest with: (a) the constraint of not knowing the reward of any effort increment, apart from the fact that it is non-negative; (b) the restriction of receiving a three-valued error measure as contest feedback; and (c) the commitment of avoiding unnecessary efforts, i.e. exceeding the adversary's, that would result in a penalty. In greater detail Bob and Alice (B and A for short) are playing the following game.

Game_1. *The game consists in a series of contrasts between B and A on random instances s drawn from a set* $\mathfrak{S}$ *(huge, but finite and discrete) with the following properties:*

* Corresponding author. Tel.: +39 02 50316284; fax: +39 02 50316228.
E-mail addresses: apolloni@dsi.unimi.it (B. Apolloni), bassis@dsi.unimi.it (S. Bassis), gaito@dsi.unimi.i (S. Gaito), malchiodi@dsi.unimi.it (D. Malchiodi), zoppis@bioinformatics.bio.disco.unimib.it (I. Zoppis).

0304-3975/$ – see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2010.02.011

- *on each s B may (i) win, (ii) lose, or (iii) tie with respect to A;*
- *each player owns an integer parameter, let us call them* strength γ_B *and* γ_A *respectively, which can take a finite number of values. According to the relation defeat* $\lhd$ *tie* $\lhd$ *victory, for a fixed value of* γ_A *the increment of B's strength does not diminish his contest results. Both for minimum* γ_A *and whatever* γ_B*, and for maximum* γ_B *and whatever* γ_A *B always either wins or ties. With each pair of strengths with intermediate relationships among them, the contest results depend on the single instances;*
- *the two players have different roles:*
  - *A maintains her strength at a fixed point;*
  - *B increases his strength by one unit whenever he loses;*
- *B's goal is to find (learn) the minimal strength that will let him lose in the future with a probability below a small threshold. His strategy is to achieve this with a good confidence level. That means he will keep playing the game until he achieves this confidence.*

We underline the peculiarity of B's goal. On the one hand, the monotony of the contest results concerns the single instances, so that with a same pair of strengths $\langle \gamma_A, \gamma_B \rangle$ B may win with some instances, tie with others and lose with the remaining ones. On the other, he does not aim at beating A in any future contest. Less ambitiously, he wants to lose in the future with low probability — an option which could allow him a smaller strength than with the former goal. Since contests are based on random instances whose distribution law is not considered by B's strategy, we will see that also the values assumed by losing probability must be dealt with in terms of random variables. This shifts the solution of the game into a problem of learning a suitable strength for B directly from the results of the contests carried out.

Learning as a tool for winning a game has been considered by many authors, most working in the field of economic research but some in neural network contexts too. The general idea is to enrich the family of Nash equilibria [1] thanks to the adaptivity of the strategies to the actual contests history. This can be achieved either by using beliefs model algorithms (such as Bayesian [2] and calibrated learning [3]) or by updating schemes that do not make any assumptions on the game structure [4,5]. Learning may be viewed as a *game against nature* [6] *per se*, where one of the opponents does not adapt his playing strategy to the contest evolution, so that the solution may lie in an extended Nash equilibrium among the reward/punishment agents of the reinforcement learning paradigm [7,8].

Game_1 belongs to the class of games against nature since B's behavior does not affect A's strategy. The single contest is in line with the *monotone game* thread [9], as a player's strength increment does not narrow his winning instance set. The peculiarity of our match lies however in the fact that the evaluation of B's actions is made through the construction of confidence intervals for the losing probability rather than in terms of optimizing the expected value of a utility function [6]. To fix ideas we might frame this game in a market competition where two factories produce similar goods (e.g. cars). At each new issue a single competitor chooses particular gadgets to be embedded in its model (e.g. a GPS or some enhanced safety devices) with the aim of attaining a degree of success more or less on par with its rival's. The monotone effort–effect mechanism characterizing the players, together with the impossibility of giving it a quantitative description, may be exemplified by an approximate solution of a knapsack problem [10] where skillful algorithms improve the approximation degree with the amount of involved computational resources [11]. It fits the original learning task where for instance the identification of neural network parameters for computing any specific function is a computationally hard problem [12] that we try to solve with consistent statistics [13]. It also fits our figuration of the market competition where a knapsack problem may underlie the inclusion of gadgets in the product budget.

We manage the game on distribution-free random instances. This is a common framework for learning algorithms that matches some unpredictability of the market solicitations and reactions as well. Through our game we frame the original subsymbolic learning problem within a modified version of the well-known Probably Approximately Correct (PAC) learning framework [14]. Indeed primary goal of the game is to compute a confidence interval for the expected value R of a loss function $L(s)$ for a given strength γ_B on the basis of a statistic drawn from a set of examples. Each example is made of pairs $\langle s, \lambda(s) \rangle$, where s is an instance of the knapsack problem and the label $\lambda(s)$ is set to 1 if B loses, otherwise to 0. The loss function $L(s)$ coincides with this label, so that R, as the mean of a Bernoulli variable, equals the probability that B loses in a subsequent contest. A distinguishing feature of this learning problem lies in the fact that the available labeling of instances changes over time, in the sense that, like in an on-line strategy, a losing instance may change into a tying or even winning one as B's strength increases. Therefore, as a further facility B may ask for the labels of the examples also in correspondence with the subsequent strength values he takes. This motivates the request for some changes in the probabilistic context supporting the learning problem (the mentioned changes on the PAC model) and new inference tools offered by the *Algorithmic Inference* framework [15–17].

We solve the learning problem underlying the game in both batch and on-line modalities. In the former we simply answer the question: "How long must a game last for B to be confident of having reached the opponent's ability, except for a very small losing probability?". We check easily that this length is polynomial in the characteristic size of the competition. In the second modality we follow the single track of our game so as to decide when to stop the game satisfactorily. The analysis of the random sequence of instances continuing those processed during the game highlights operational conditions giving rise to wide confidence intervals for the defeat probability even after a long game. We deepen the dynamics of the processes underlying these games in terms of collected statistics and the efficiency of B strategy. Returning to subsymbolic learning processes, we discuss similar dynamics basing on the course of the cost function minimized in a training session. So we get dynamical features for deciding to stop the session without necessity of a testing phase.

The paper is organized as follows. In Section 2 we discuss our statistical framework, introducing the twisting argument and reformulating the PAC learning problem in this framework. In Sections 3 and 4 we solve the game inference problem in the two (batch and on-line) modalities. In Section 5 we discuss solutions and their extension to the original learning task with the help of numerical experiments. The last section is devoted to outlooks and concluding remarks.

2. The statistical framework

In this section we will briefly sum up the inference approach we use to accomplish a learning task. Nevertheless this section takes up almost one third of the article. The point is that we use a really new statistical framework to face the above game, where no fixed probability distribution is assumed in advance for the results of the contests, neither in the relaxed hypothesis that this distribution remains unknown. This perspective is not new *per se*, since it dates back to Laplace [18], Fisher [19], Tukey [20] and in more recent times De Finetti [21]. We reread it in terms of the Algorithmic Inference theory [15] in order to finalize results to the learning problem. Readers already acquainted with this theory may skip this section; those wanting to learn about the subject matter in greater depth may refer to the book [15].

Right from the start, the object of our inference is a string of data $\mathbf{x}$ (possibly of infinite length) that we partition into a prefix we assume to be known at present (and therefore call *sample*) and a suffix of unknown future data we call a *population*. Here, the terms sample and population have the same meaning as in classical statistical frameworks, from which we also draw the usual notations. Therefore, by default capital letters (such as U, X) will denote random variables and small letters (u, x) their corresponding specifications; the sets the specifications belong to will be denoted by capital gothic letters ($\mathfrak{U}, \mathfrak{X}$), while strings of these specifications are in boldtype ($\mathbf{u}, \mathbf{x}$). With these notations, a sample $\{x_1, \ldots, x_m\}$ is a set of specifications of the random variable X, where m is denoted as the *size* of the sample. A population from X is a sample of possibly infinite size representing a set of X specifications that we may expect to observe in the future.

All these data share the feature of being independent observations of the same phenomenon. Therefore without loss of generality we assume these data to be the output of some function g_ϑ having input from a set of independent random variables U uniformly distributed in the unit interval — effectively, the most essential source of randomness. The probability integral transformation theorem [22] ensures that at least one g_ϑ always exists for every random variable X; it consists of the inverse of the X cumulative distribution law for X continuous with obvious extensions for X discrete.

2.1. The twisting argument

We will refer to $\mathcal{M} = (U, g_\vartheta)$ as a *sampling mechanism* and to g_ϑ as an *explaining function*. This function is precisely the object of our inference. Let us consider, for instance, the sample mechanism $\mathcal{M} = (U, g_p)$, where U is the above uniform random variable and

$$g_p(u) = \begin{cases} 1 & \text{if } u \leq p \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

maps from U specifications to specifications of a Bernoulli random variable X of mean p. As can be seen from Fig. 1, this function can be used to obtain both sample and population items of $\mathbf{x}$, and with a given sequence for u we obtain different binary strings depending on the height p of the threshold line. This is the true reason why we may infer something about unseen population from an observed sample. Thus for each fixed k and $\widetilde{p}$ it is easy to derive the following implication chain

$$\left(k_{\widetilde{p}} \geq k\right) \Leftarrow (p < \widetilde{p}) \Leftarrow \left(k_{\widetilde{p}} \geq k + 1\right) \tag{2}$$

where k is the number of 1 observed in a sample of size m, and $k_{\widetilde{p}}$, as a specification of the random variable $K_{\widetilde{p}}$, counts the number of 1 in the sample if the threshold in the explaining function switches to $\widetilde{p}$ *for the same specifications* of U. The consequent bounds on the probability

$$\mathrm{P}\left(K_{\widetilde{p}} \geq k\right) \geq \mathrm{P}\left(P < \widetilde{p}\right) = F_P\left(\widetilde{p}\right) \geq \mathrm{P}\left(K_{\widetilde{p}} \geq k + 1\right) \tag{3}$$

characterize the cumulative distribution function (c.d.f.) F_P of the parameter P, having denoted with $\mathrm{P}(A)$ the probability of the event A. In our statistical framework indeed the above height p is a specification of a continuous random variable P in $[0, 1]$. This variable represents the asymptotic ($M \to +\infty$ in Fig. 1) frequency of 1 in the population that is compatible, as a function of a suffix of the U sample, with the number k of actually observed 1 over the m bits in the X sample. In words, the implications in (2) say that a raising of the threshold line that increments the number of 1 in the population cannot decrement the number of 1 in the sample and *vice versa*. Eq. (3) comes straight from marginalizing [13] the joint U's distribution in respect to the population when we deal with sample statistic $K_{\widetilde{p}}$ and to the sample when we deal with population parameter P.[1]

[1] This is the true connection with the Kolmogorov probabilistic framework. We start from the probability space $(\Omega, \Sigma, \mathrm{P})$, where Ω is the unitary segment, Σ the related sigma-algebra and P is uniformly distributed on Ω. On this space we define the random variable U, the rest comes from functions of this variable.

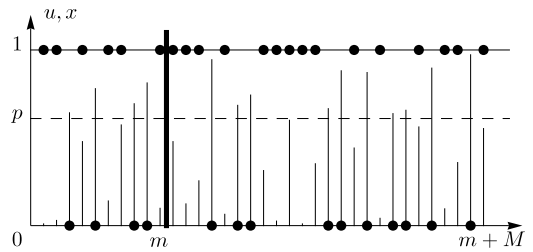

Fig. 1. Generating a set of Bernoulli variable specifications $\{x_1, \ldots, x_{m+M}\}$. Horizontal axis: index of the specifications; vertical axis: both u (lines) and x (bullets) values. The bold vertical line separates a sample of size m from a population of size M. The horizontal dashed line of height p realizes a mapping from U to X through (1).

Note the asymmetry in the implications. It derives from the fact that:

- raising the threshold parameter in g_p cannot decrease the number of 1 in the observed sample, but
- we can recognize that such a raising occurred only if we really see a number of 1 in the sample greater than k.

We will refer to any expression similar to (2) as a *twisting argument*, since it allows us to exchange events on parameters with events on statistics.

2.1.1. Confidence interval for the parameter of a Bernoulli variable

Relations shown in (3) enable us to compute confidence intervals for P. In our inference scenario indeed, in place of a *specific* random variable as given, we refer to a *family* of random variables with free parameters whose distribution law is discovered through a twisting argument. For short, we denote this family as *a given* random variable, say X, yet with *random* parameters.

Definition 1. Given a random variable with a free parameter L and a real number $0 \leq \delta \leq 1$, (l_i, l_s) is called a $1-\delta$ *confidence interval* for L if

$$\mathrm{P}(l_i < L < l_s) = 1 - \delta. \tag{4}$$

The quantity δ is called the *confidence level*. We denote as *tail symmetric* a confidence interval such that $P(L > l_i) = P(L < l_s)$.

Lemma 2. *Let X denote a random variable distributed according to a Bernoulli law of mean P, $\{x_1, \ldots, x_m\}$ a sample and $k = \sum_{i=1}^{m} x_i$ the number of 1 in it. A tail symmetric confidence interval of a level less or equal δ for P is (l_i, l_s) where l_i is the $\delta/2$ quantile of the Beta distribution of parameters k and $m - k + 1$, and l_s is the analogous $1 - \delta/2$ quantile for parameters $k + 1$ and $m - k$.*

Proof. Consider the probability relation (3). Since $K_{\widetilde{p}}$ follows a Binomial distribution law of parameters m and $\widetilde{p}$, it reads

$$\sum_{i=k}^{m} \binom{m}{i} \widetilde{p}^i (1-\widetilde{p})^{m-i} \geq F_P(\widetilde{p}) \geq \sum_{i=k+1}^{m} \binom{m}{i} \widetilde{p}^i (1-\widetilde{p})^{m-i}. \tag{5}$$

Having introduced the Incomplete Beta function I_β as the c.d.f. of the random variable $Z_{h,r}$ following a Beta distribution of parameters h and r [23], that is

$$I_\beta(h, r) \equiv \mathrm{P}(Z_{h,r} \leq \beta) = 1 - \sum_{i=0}^{h-1} \binom{h+r-1}{i} \beta^i (1-\beta)^{h+r-1-i}$$

the above bounds can be written as

$$I_{\widetilde{p}}(k, m-k+1) \geq F_P(\widetilde{p}) \geq I_{\widetilde{p}}(k+1, m-k).$$

Eq. (4) can be solved by dividing the probability measure outside (l_i, l_s) in two equal parts in order to obtain a two-sided interval symmetric in the tail probabilities. Getting

$$\Delta I = I_{l_s}(k+1, m-k) - I_{l_i}(k, m-k+1)$$

as a lower bound to $F_P(l_s) - F_P(l_i) = \mathrm{P}(l_i < P < l_s)$, we obtain the extremes of the interval as the solutions l_i and l_s of the system of equations

$$I_{l_s}(k+1, m-k) = 1 - \delta/2 \tag{6}$$
$$I_{l_i}(k, m-k+1) = \delta/2. \quad \square \tag{7}$$

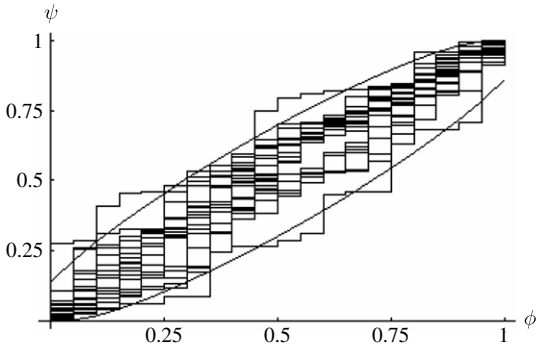

Fig. 2. Generating 0.9 confidence intervals for the mean P of a Bernoulli variable. $\phi = k/m$: frequency of 1's in the sample; $\psi = h/M$: frequency of 1's in the population; $m = 20$ and $M = 200$. Stepwise lines: $\langle \phi, \psi \rangle$ trajectories; curves: course of the confidence interval extremes.

We visualize the implications substantiating the twisting argument (2) and appreciate their probabilistic consequences in Fig. 2. In order to draw one of the stepwise lines filling the picture, let us consider a string of $20 + 200$ unitary uniform variables representing, respectively, the randomness source of a sample and a population of Bernoulli variables. Then according to the explaining function (1) we compute a sequence of Bernoullian 220 bits long vectors with p rising from 0 to 1. The upper-left corners of the steps in the questioned lines have coordinates $k/20$ and $h/200$, accounting the pairs of 1's frequencies in the sample and in the population, respectively, that jointly occur in the above vectors. By repeating this experiment 20 times (each time using different vectors of uniform variables) we obtained the bundle of lines in figure. Then we drew on the same graph the solutions l_i and l_s of Eqs. (6)–(7) that we obtained with $\delta = 0.1$ and k ranging from 0 to 20. As we can see, for a given value of k the intercepts of the above curves with a vertical line with abscissa $k/20$ determine an interval containing almost all intercepts of the steps with the same line. A more intensive experiment would show that, in the approximation of $h/200$ with the asymptotic frequency of ones in the suffixes of the first 20 sampled values, on all samples (and even for each sample if we draw many suffixes of the same one), almost $100(1 - \delta)\%$ of the step upper-left corners fall within the analytically computed curves. This represents the operational counterpart of Definition 1, since the asymptotic frequencies are the specifications of the random parameter P characterizing the Bernoulli variable X as a suffix of the observed sample and $(1 - \delta)$ is the questioned probability of finding it within the confidence interval (l_i, l_s).

2.1.2. Toward more complex random variables

An extension of (2) to a generic sampling mechanism (U, g_ϑ), where g_ϑ explains a sequence of random variables X having a free parameter ϑ, may be stated as follows. Denoting $\mathbf{x} = \{x_1, \ldots, x_m\}$ a specific sample of X, let $\mathbf{u} = (u_1, \ldots, u_m)$ be a m-ple of values in $[0, 1]$ such that $g_\vartheta(\mathbf{u}) = (g_\vartheta(u_1), \ldots, g_\vartheta(u_m)) = (x_1, \ldots, x_m)$. We look for a statistic T such that if $T(x_1, \ldots, x_m) = t$ then

$$(T(g_{\widetilde{\vartheta}}(\mathbf{u})) \geq t + \mu_1) \Leftarrow (\vartheta < \widetilde{\vartheta}) \Leftarrow (T(g_{\widetilde{\vartheta}}(\mathbf{u})) \geq t + \mu_2) \tag{8}$$

for almost every $\mathbf{x}$ and suitable constants $\mu_1, \mu_2 \geq 0$ with $\mu_1 \leq \mu_2$.

In the following we will simply refer to the property T constituted by the number of sample elements satisfying a given condition, as the target of our inference will be the probability measure of the support of special Boolean functions. However, in a more general perspective elsewhere we suggest working with sufficient statistics [15].

In order to extend the proposed inference model to the measure of the mentioned Boolean domains, consider a random variable Y in a space $\mathfrak{Y}$ (explained by a proper sampling mechanism $\mathcal{M}$); for any fixed subset $c \subset \mathfrak{Y}$, we may refer to the random variable $X' = 1$ if $Y \in c$ else 0. This is another way of explaining a Bernoulli variable whose behavior is identical to that of X introduced through (1), provided that $\mathrm{P}(U \leq p) = \mathrm{P}(Y \in c)$.

The situation is different if c too is unknown. In this case we have two ways of moving p to $\widetilde{p}$ in (2) starting from an assigned shape h (call it hypothesis) to c. We can both raise the probability of falling into h and enlarge h. To extend right implication in (2) we focus on the second way. Thus, for a whatever fixed parameterization of Y distribution law, we obtain a similar picture as in Fig. 1 for X' considering an expanding sequence $B(h) = (B_1, B_2, B_3, \ldots \text{ s.t. } B_i \subseteq B_{i+1}, \forall i)$ of subsets of $\mathfrak{Y}$ pivoted around h in the sense that h belongs to $B(h)$. For the mere sake of simplicity, in Fig. 3 we assumed B_i to have the same circular shape as h, since this shape is immaterial. Moving from one element of $B(h)$ to another corresponds to raising or lowering the threshold line in Fig. 1, with identical implication chain in regard to the number of 0 and 1 in sample and population (namely, *if* the number of 1 observed in the sample increases, *then* the number of analogous elements in the population will increase). Now, let us refer h to the set of uniformly distributed variables in input to the function explaining Y (for instance assume $\mathfrak{Y} = \mathbb{R}^2$ and a pair (U_1, U_2) generating the coordinates $(g_{\vartheta_1}(U_1), g_{\vartheta_2}(U_2))$ of Y). In this framework we can reconsider the circle sequence in the picture in Fig. 3 as mappings of the same h with changes of parameters $(\vartheta_1, \vartheta_2)$ increasing $\mathrm{P}(Y \in h)$. Thus, if we put together the two ways of augmenting $p = \mathrm{P}(Y \in h)$, the condition $p < \widetilde{p}$ implies the existence of a domain of *exactly* measure $\widetilde{p}$ including h.

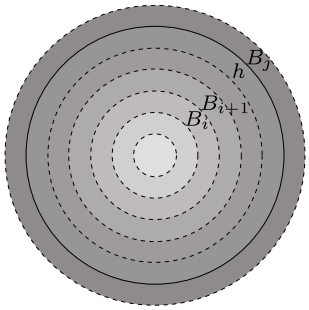

Fig. 3. A nested sequence of domains B_i (dashed circles) in $\mathbb{R}^2$ having a given domain h (plain circle) among its elements.

To operationally exploit these arguments we are left with the problem of: (i) how to recognize that h is the pivot of a sequence having among its elements a $B_{\tilde{i}}$ of measure $\widetilde{p}$ including h (right implication in (8)), and (ii) how to relate the necessary condition to a property of the sample (left implication in (8)).

2.2. A special perspective on probabilistic learning

2.2.1. Inferring rectangle measures

Let us move from the unidimensional case of Fig. 1 to a bidimensional case like in Fig. 4(a).

Here again we give label 1 to the single coordinate $u_j, j = 1, 2$ (each ruled by a corresponding uniform random variable U_j) if it falls below a given threshold p_j, label 0 otherwise. Moreover we give to the point $\mathbf{a}_i$ of coordinates (u_1, u_2) a label equal to the product of the labels of the single coordinates. Thus the probability p_c that a point **A** falls in the open rectangle c bounded by the coordinate axes and the two mentioned threshold lines (for short we will henceforth refer to these rectangles as *b_rectangles*) receiving label 1 is $p_1 p_2$. Let us complicate the inference scheme in the two ways mentioned in the previous section.

(1) We move from U_j to the family of uniform random variables W_j in $[0, \vartheta]$ explained by the function $w = \vartheta u$ with $\vartheta \in (0, +\infty)$.
(2) We maintain the same labeling rule but do not know c, i.e. the thresholds p_1 and p_2. Rather, on the basis of a sample of W, within the class of b_rectangles containing all 1-labeled sample points yet excluding all 0-labeled sample points (consistent b_rectangles as statistics), we will identify it with a maximal one (call it h), i.e. a one that cannot be included in another element of the class (hence having edges just before a pair of 0-labeled points closest to the axes' origin). Letting p'_1 and p'_2 be the length of these edges, we presently look for the probability $p_h = p'_1 p'_2 / \vartheta^2$ representing the asymptotic frequency with which future points **w** (with coordinates (w_1, w_2) generated with the above sampling mechanism for any ϑ) will fall in h.

To exploit the above arguments about the abstract $B(h)$ sequence we must realize that this sequence is actually pivoted around the hypothesis h we have built through our algorithm. Said in other words, we may imagine a whole family of B sequences, each pivoted on a possible rectangle. In some sequences the domain $B_{\tilde{i}}$ of measure $\widetilde{p}$ will include the pivot in other ones is included by it. Thus we need witnesses that for our actual h computed from the actual sample the pivot is included in $B_{\tilde{i}}$. But this happens if two special points – in Fig. 4(a) exactly the marked negative **w** preventing the rectangle expanding on the left and the marked negative **w** preventing the rectangle expanding on the up – are included in $B_{\tilde{i}}$. Indeed, having one point binding the h vertical edge is not sufficient for ensuring that the h horizontal edge does not trespass the $B_{\tilde{i}}$ contour, and *vice versa*. Thus let us enrich the family of sequences having for each rectangle and each possible pair of witness points the pivot constituted by the union of the rectangle with these points. In respect to the sequence pivoted on our actual rectangle and witnesses we have that if 2 or more negative points are included in $B_{\tilde{i}}$ for sure the witness points are among them. Hence a twisting argument reads:

$$(p_h < \widetilde{p}) \Leftarrow \left(k_{\widetilde{p}} \geq k + 2\right) \tag{9}$$

where k is the number of 1-labeled points, $k_{\widetilde{p}}$ is still a specification of a Binomial random variable of parameters m (sample size) and $\widetilde{p}$, accounting for the sample points contained in $B_{\tilde{i}}$.

From the left, let us consider the family of b_rectangles. As ϑ is free, $(p_h < \widetilde{p})$ requires that there must be an enlargement of h whose measure for a proper ϑ exactly equals $\widetilde{p}$. But since both edges of h are bounded by a negative point, this enlargement must contain at least one point more than h itself. Formally

$$\left(k_{\widetilde{p}} \geq k + 1\right) \Leftarrow (p_h < \widetilde{p})\,. \tag{10}$$

Putting together the two pieces of twisting argument, we obtain the corresponding bounds on probabilities as follows:

$$\mathrm{P}\left(K_{\widetilde{p}} \geq k + 1\right) \geq \mathrm{P}\left(P_h < \widetilde{p}\right) = F_{P_h}\left(\widetilde{p}\right) \geq \mathrm{P}\left(K_{\widetilde{p}} \geq k + 2\right). \tag{11}$$

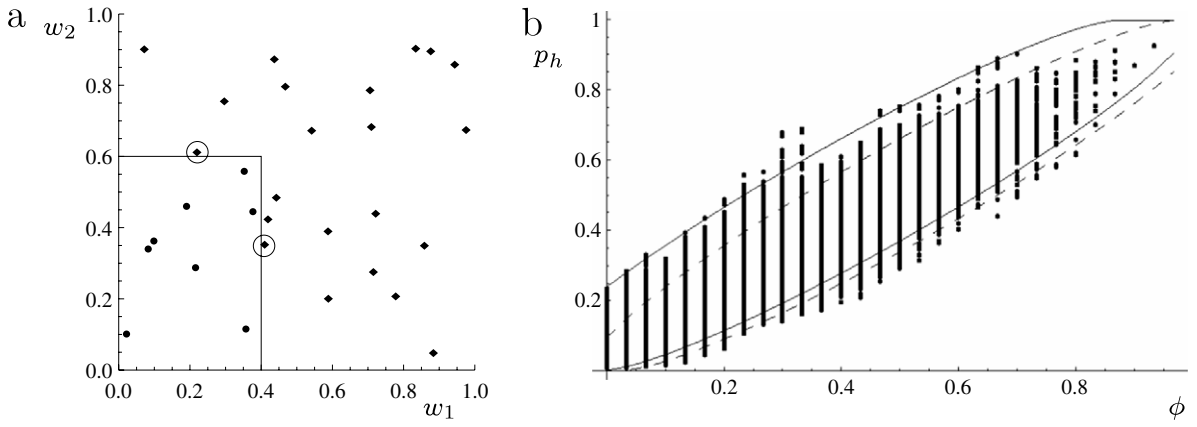

Fig. 4. Generating 0.9 confidence intervals for the probability P_h of a b_rectangle in $\mathfrak{Y} = [0, 1]^4$ from a sample of 30 elements. (a) The drawn sample and one of its possible labelings in a two-dimensional projection; points inside circles are deputed sentinels. (b) Points: courses of the frequency ϕ and probability p_h of falling inside a b_rectangle with a lattice of labeling functions. Plain curves: trajectories described by the confidence interval extremes for the actual four-dimensional case. Dashed curves: trajectories described by the confidence interval extremes computed from Lemma 2.

Generalizing our arguments to an n-dimensional space, Lemma 2 extend to the following:

Lemma 3. *Let $\mathfrak{Y} = \mathbb{R}^n$, $((Y_i, X_i), i = 1, \ldots, m, \ldots)$ be an infinite sequence of pairs of random variables drawn from the Cartesian product space $\mathfrak{Y} \times \{0, 1\}$. Let us assume that for each specification of the sequence a b_rectangle c exists in $\mathfrak{Y}$ such that*

$$x_i = \begin{cases} 1 & \text{if } y_i \in c \\ 0 & \text{otherwise} \end{cases} \tag{12}$$

for each i, and h denote a maximal b_rectangle with respect to the sample constituted by the first m elements in the sequence. For $k = \sum_{i=1}^{m} x_i$, a tail symmetric confidence interval of level δ for $P_h = \mathrm{P}(Y \in h)$ is (l_i, l_s) where l_i is the $\delta/2$ quantile of the Beta distribution of parameters $k + 1$ and $m - k$, and l_s is the analogous $1 - \delta/2$ quantile for parameters $k + n$ and $m - k - n + 1$.

Proof. It is easy to realize that in this more general case at most n witness points are necessary, for whatever probability distribution on Y[2]: so we just rewrite (11) in terms of a Beta distribution law after substituting 2 with n in the right implication threshold:

$$I_\alpha(k + 1, m - k) \geq F_{P_h}(\alpha) \geq I_\alpha(k + n, m - k - n + 1) \tag{13}$$

and use the same arguments as in Lemma 2. □

We extended the experiment shown in Fig. 2 to the new framework as follows. We consider 20 sampling histories. Each sample is made of $m = 30$ labeled points $\mathbf{w}_i$ of a four-dimensional continuous unitary cube $\mathfrak{Y}$. We chose this dimension to mark the drift from the Bernoulli variable, whereas the rectangle in Fig. 4(a) represents just a projection in a two-dimensional space. To consider a non-flat probability distribution, the sampling mechanism of the jth coordinate of $\mathbf{w}$ has explaining function $g(u) = u^j$.[3] Then, to figure out a wide set of possible labeling mechanisms (12), we stored the labels attributed to sample points from all b_rectangles within the unitary hypercube with vertices in a grid of edge 1/10 on each dimension. Then we forgot the source figures and for each labeling computed a maximal consistent b_rectangle h, and we drew the graph in Fig. 4(b). Namely, for each sample and each labeling according to the above rectangles we reported on the graph the actual frequency ϕ and the probability p_h (analytically computed on the basis of the rectangle edges) of having a point in $\mathfrak{Y}$ belonging to the guessed maximal hypothesis. On the same graph we also reported the curves describing the course of the tail symmetric 0.9 confidence interval for P_h with the observed frequency of falling inside the rectangle h according to Lemma 3 with $n = 4$. Finally, for the sake of comparison we also drew the curves obtained from Lemma 2 for a pure Bernoulli variable. In spite of some apparent unbalancing in the figure, the percentages of points $\mathbf{w}_i$ falling out of upper and lower bound curves are approximately equal, 3.32% and 3.67% respectively. Thus they satisfy the 5% upper bounds (corresponding to $\delta/2$, where δ is the confidence level of the interval) used for drawing these curves. The analogous percentages, 16.2% and 0.28%, denote the inadequacy of the curves drawn for the Bernoulli distribution.

2 The interested reader can see a more detailed proof in [24,25].

3 Hence we adopt a different sampling mechanism than the one used for simply introducing our first learning problem. This is not a drawback as the claim of Lemma 3 is distribution-free.

2.2.2. Learning a concept class

Definition 4. For a given space $\mathfrak{X}$, a *concept* is a Boolean function $c : \mathfrak{X} \mapsto \{0, 1\}$. A set of concepts defines a *concept class* C.

By abuse of notation we will refer to a concept either as a function c or as the support of it.[4]

Definition 5. Having fixed a space $\mathfrak{X}$, consider a random variable X and a concept c on it. An *example* Z of c is a pair $(X, c(X))$. This means that an example associates to an $x \in \mathfrak{X}$ a label $c(x)$ whose value is 1 if x belongs to c and 0 otherwise. *Vice versa* another concept h on $\mathfrak{X}$ is consistent with an example $z = (x, c(x))$ of c if $h(x) = c(x)$. The function h is consistent with a set of examples if it is consistent with all its elements.

Given $\mathfrak{X}$ and general properties that define a class C of concepts on it, we consider the learning task of inferring from a set of examples the concept c in C underlying their labels. Actually we are satisfied with identifying in a set H a concept that we call *hypothesis* h, approximating c well. As mentioned before, to appreciate how close h is to c we refer to the probability distribution on $\mathfrak{X}$. We do not know it explicitly, but we observe the examples, and with a *compatible* distribution we will be questioned about $c(x)$ on new inputs. Our answer will be wrong on those points that either belong to c and not to h (thus we will answer 0 using $h(x)$ while the correct answer is $c(x) = 1$) or belong to h and not to c (and our answer will be $h(x) = 1$ in place of $c(x) = 0$). The set of these points denotes the symmetric difference $c \div h$ between c and h, and we quantify the approximating capability of h through the probability $U_{c \div h} = \mathrm{P}(c \div h)$ of this region.

A *learning algorithm* is a procedure $\mathcal{A}$ to generate a family of hypotheses h_m with their respective $U_{c \div h_m}$ converging to 0 in probability with increasing size m of the example set given in input to the procedure [14]. For instance, in Fig. 5(a) the set $\mathfrak{X}$ of the experimental outcomes coincides with the Cartesian plane; the learning task is to identify one particular circle c within the concept class C of all possible circles in the plane. This might be a mathematical model for identifying the site and the emission range of a source of radiating pollution, such as noise, X-ray and so on, in a flat homogeneous region. The set of examples might be identified with a set of monitoring stations taking label 1 if pollution is detected above a given threshold. We are concerned with the probability that say, a Mr. John Smith is exposed to radiation.

We formally frame this task as follows.

Definition 6. For fixed $m \in \mathbb{N}$, a *labeled sample* is a set

$$\mathbf{z}_m = \{(x_i, b_i),\ i = 1, \ldots, m\}$$

where b_i are Boolean variables and $x_i \in \mathfrak{X}$. Given a $\mathbf{z}_m$ and a concept class C, assume that for every $M \in \mathbb{N}$ and every (labeled) population $\mathbf{z}_M$ generated with the same sampling mechanism $\mathcal{M}$ of $\mathbf{z}_m$ a c exists in C explaining both sample and population labels, i.e. such that $\mathbf{z}_{m+M} = \{(x_i, c(x_i)),\ i = 1, \ldots, m + M\}$. For a set $\mathfrak{Z}_m$ of $\mathbf{z}_m$s sharing C, we call *learning algorithm* an algorithm $\mathcal{A} : \mathfrak{Z}_m \mapsto \mathsf{H}$, where H is a concept class (possibly different from C) that we specifically call class of hypotheses, which

- for any $\mathbf{z}_m \in \mathfrak{Z}_m$;
- for some pair[5] of parameters $\varepsilon, \delta > 0$; (denoted henceforth as *accuracy* parameters)

computes a function, that we denote as *hypothesis* h, such that the confidence interval $(0, \varepsilon)$ for the measure $U_{c \div h}$ of the symmetric difference between h and the explaining functions of populations compatible with $\mathbf{z}_m$ has at least confidence $1 - \delta$ (see Fig. 5(a)).

In formulas:

$$\mathrm{P}(U_{c \div h} \leq \varepsilon) \geq 1 - \delta. \tag{14}$$

It is evident that the probability measure of the symmetric difference is a random variable, too. The way of interpreting its randomness represents the divide between conventional approaches and our own.

Coming back to the initial example of localizing a polluted region, we assume in the conventional approach that the inhabitants distribution law is available (at least in terms of the list from which to randomly pick a sample, say the local telephone directory) and that a circle c exists delimiting the polluted region. Then we extract a sample – the monitoring stations – receiving labels from it and must approximate c with another circle h computed on the basis of the observed labeled points. To address the computation, we may assume for instance the consistency constraint of h giving the same labels as c to the sampled points. Since h is a function of the random sample, it is a random variable itself, having specifications in $h_0, h_1, \ldots$ as in Fig. 5(b), whose probability law derives from the sample's. In the figure h_0 is consistent with the points drawn therein, while we may assume the other hypotheses consistent with other sets of sampled points each.

In our perspective we dually consider that a circle h is computed on the basis of the observed data (for instance with the same consistency constraint) and we assume that a circle c will exist satisfying the same constraints on both the actual and future observations. Since we can have plenty of different observation histories we can have a lot of different circles,

[4] i.e. the set of points $x \in \mathfrak{X}$ such that $c(x) = 1$.

[5] Elsewhere a tighter request is that for every (ε, δ) an m_0 exists such that for every $m > m_0$ the claim of the definition holds.

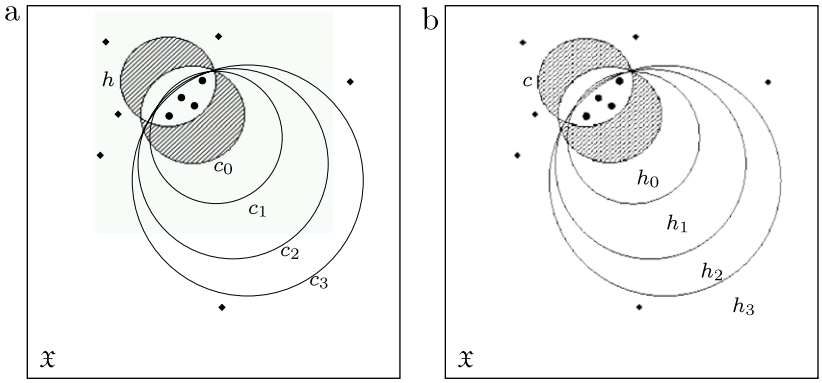

Fig. 5. Algorithmic and conventional inferences in computational learning. $\mathfrak{X}$: the set of points belonging to the Cartesian plane; bullets and diamonds as in Fig. 4(a). Line filled region: symmetric difference between candidate concept and hypothesis. (a) Algorithmic inference framework. h: the circle describing the sample according to a learning algorithm; c_i: possible circles describing the population. (b) The corresponding conventional learning framework. c: a concept from the class of circles; h_i: hypotheses approximating c.

like c_0, c_1, ... as in Fig. 5(a), as well. However, the fact that both c and h must give the same labels to the actual sample allows us to probabilistically describe the family of possible c's in terms of the mentioned error probability, provided that no exceptional demographic phenomenon makes the sites of the new requests on the pollution state heterogeneous in respect to the monitoring stations (i.e. provided that the sampling mechanism is the same). Thus all concepts must be consistent exactly with the points drawn in Fig. 5(a). From an operational point of view, the difference between the two approaches lies in the fact that in the former we look for a c that labels every point of a fixed population $\mathbf{z}_M$ and try to discover it (rather, its approximation h) in spite of samples' variability (as specifications of $\mathbf{Z}_m$). Here we assume a dual perspective: for fixed sample $\mathbf{z}_m$ we can have different populations (as specifications of $\mathbf{Z}_M$), hence different c's explaining them.

2.2.3. Sentry points

Like in the rectangle case study, we relate the measure of the symmetric difference to the number of points falling inside it plus those outside it necessary for witnessing the inclusion of $c \div h$ into a domain B_{i_ε} of measure ε within a sequence $B(c \div h)$ as in Section 2.1.2, for a possibly very small ε. We dually characterize the functionality of these points – which we call either *sentinels* or *sentry points* and *frontier* as a whole – in terms of preventing a given concept c from being fully included by another concept within C. Like sentinels of a fortress, they are located in such sites that, given his own topological constraints, an invader may never elude them all in surrounding the town-walls. Namely, they are assigned by a *sentineling function* $\mathbf{S}$ (whose formal definition is given in [24]) to each concept of a class in such a way that:

- they are external to the concept c to be sentineled and internal to at least one other including it,
- each concept c' including c has at least one of the sentry points of c either in the gap between c and c' or outside of c' and distinct from the sentry points of c', and
- they constitute a minimal set with these properties.

The frontier size of the most expensive concept to be sentineled with the least efficient sentineling function, i.e. the quantity $\mathsf{D}_\mathsf{C} = \sup_{\mathbf{S},c} \#\mathbf{S}(c)$, is called *detail* of C, where $\mathbf{S}$ spans also over sentineling functions on subsets of $\mathfrak{X}$ sentineling in this case the intersections of the concepts with these subsets. Moving to symmetric differences, for another set H of concepts let us consider the class of symmetric differences $c \div \mathsf{H} = \{c \div h \ \forall h \in \mathsf{H}\}$ for any c belonging to C. The detail of a concept class H w.r.t. c is the quantity $\mathsf{D}_{c,\mathsf{H}} = \mathsf{D}_{c \div \mathsf{H}}$. *Vice versa*, for a fixed h we consider an augmented sentineling functionality against the entire class C where to each h is assigned a minimal set of points necessary to sentinel any $c \div h$, for any $c \in \mathsf{C}$ against $c \div \mathsf{H}$. Let us denote by $\mathsf{D}_{(\mathsf{C},\mathsf{H})_h}$ the cardinality of the sentry set of h, and define the overall detail of the class $\mathsf{C} \div \mathsf{H} = \cup_{c \in \mathsf{C}} c \div \mathsf{H}$ as the quantity $\mathsf{D}_{\mathsf{C},\mathsf{H}} = \sup_{h \in \mathsf{H}} \{\mathsf{D}_{(\mathsf{C},\mathsf{H})_h}\}$.

Example 7.
- With reference to Fig. 6(a), $\{x_1, x_2, x_3\}$ is a candidate frontier of c_0 against c_1, c_2, c_3, c_4. All points are in the gap between a c_i and c_0. They avoid inclusion of $c_0 \cup \{x_1, x_2, x_3\}$ in c_3, provided that these points are not used by the latter for sentineling itself against other concepts. *Vice versa* we expect that c_1 uses x_1 and x_3 as its own sentinels, c_2 uses x_2 and x_3, and c_4 uses x_1 and x_2 analogously. Point x_4 is not allowed as a c_0 sentry point since, like some diplomatic seat, should be located out of any other concepts just to avoid that it is occupied in case of c_0 invasion.
- D_C is 1 if the elements of C are oriented half-lines on $\mathbb{R}$, and 2 if they are segments on the same set.
- The class C of circles in $\mathbb{R}^2$ has detail $\mathsf{D}_\mathsf{C} = 2$, as visible in Fig. 6(b).
- D_C is $\lceil 4/3k \rceil$ if the concepts are convex polygons on $\mathbb{R}^2$ having exactly k edges (Fig. 6(c) shows a worst case frontier for the case $k = 5$; see [25] for the general case proof).
- The class of monotone clauses [26] has detail 1. Each clause $(v_i + v_j + \cdots + v_k)$ is sentineled by the point $\mathbf{x}$ of the Boolean hypercube whose ith component is 0 if v_i is an addend in the clause, 1 elsewhere.

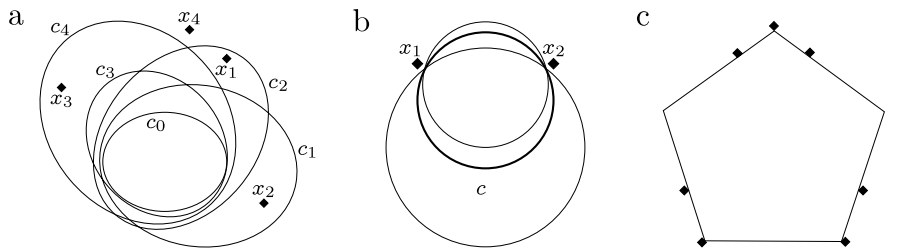

Fig. 6. (a) A schematic outlook of sentineling functionality. c_0: concept sentineled against c_1, c_2, c_3, c_4; $\{x_1, x_2, x_3\}$: c_0 frontier; (b) Two points x_1, x_2 outside c are sufficient to prevent a larger circle not containing them from including it; (c) seven points do the same in a pentagon in the worst case.

Remark 8. Note that the class we want to sentinel in the task in Fig. 5 is constituted by symmetric differences between circles. In this respect examples are all negative points in case of consistent hypotheses; some of these will act as sentry points. The worst case is when either $c \subseteq h$ or *vice versa*. In both cases it is a matter of sentineling a circle again, thus also the detail $D_{\mathsf{C},\mathsf{C}}$ of this class equals 2.

As in the case of other class complexity indices [27], the detail of a class C is a parameter extremely meaningful but difficult to compute. The interested reader can find further examples in [24,25]. For a general framing, our index proves of the same order of the well-known Vapnik–Chervonenkis dimension of a concept class [24].

2.2.4. A twisting argument for learning

Let us focus on $U_{c \div h}$ distribution law. The related statistic $k_{u_{c \div h}}$ counts the number of actual sample points falling in $c \div h$, and k_ε the analogous number for an enlargement $B_{i_\varepsilon} \supseteq c \div h$ of measure ε. The sentry points within the labeled sample actually witness, in a number of at most $\mathrm{D}_{(\mathsf{C},\mathsf{H})_h}$, the inclusion of $c \div h$ in B_{i_ε} as in (9). Considerations on left implication in (8) are the same as in (10). Thus we specify (8) as follows:

$$\left(k_\varepsilon \geq k_{u_{c \div h}} + 1\right) \Leftarrow (u_{c \div h} < \varepsilon) \Leftarrow \left(k_\varepsilon \geq k_{u_{c \div h}} + \mathrm{D}_{(\mathsf{C},\mathsf{H})_h}\right). \tag{15}$$

The companion probability chain reads:

$$I_\varepsilon(k_{u_{c \div h}} + 1, m - k_{u_{c \div h}}) \geq F_{U_{c \div h}}(\varepsilon) \geq I_\varepsilon(k_{u_{c \div h}} + \mathrm{D}_{(\mathsf{C},\mathsf{H})_h}, m - k_{u_{c \div h}} - \mathrm{D}_{(\mathsf{C},\mathsf{H})_h} + 1). \tag{16}$$

The main lesson we draw from the above discussion is that when we want to infer a function we must divide the available examples in two categories, the relevant ones and the mass. As in a professor's lecture, some, the former, fix the ideas, thus binding the difference between concept and hypothesis. The latter are redundant; but if we produce a lot of examples we are confident that a sufficient number of those belonging to the first category will have been exhibited. A series of results [24,28] allows us to state a relation between the two categories' sizes.

2.2.5. Learning modalities

A main difference in learning Boolean functions in a symbolic or subsymbolic way lies in the consistency of the hypotheses. In the former modality, as you know the shape of the target concepts you may manage for hypotheses compliant with the labels of the examples, apart a few controlled exceptions you may decide for sake of computational costs. If you may change such hypotheses in an incremental way with the number of processed examples you speak of on-line learning [29]. As a result, you have lemmas like the following:

Lemma 9 (*[24]*). *For*

- *a space* $\mathfrak{X}$ *and any probability measure* P *on it* [6]*;*
- *any concept classes* H *and* C *on* $\mathfrak{X}$*;*
- *any fairly strongly surjective*[7] *function* $\mathcal{A} : \{\mathbf{z}_m\} \mapsto \mathsf{H}$*;*
- *a labeled sample* $\mathbf{z}_m$ *from* $\mathfrak{X} \times \{0, 1\}$*;*
- *any pair* $0 < \varepsilon, \delta < 1$*;*

if

- *for any infinite suffix* $\mathbf{z}_M$ *of* $\mathbf{z}_m$ *a* $c \in \mathsf{C}$ *exists computing the example labels of the whole suffix;*
- $h = \mathcal{A}(\mathbf{z}_m)$ *has detail* $\mathrm{D}_{(\mathsf{C},\mathsf{H})_h} = \mu_h$ *and misclassify* t_h *points of* $\mathbf{z}_m$*;*
- $m \geq \max\{\frac{2}{\varepsilon \ln(1/\delta)}, \frac{5.5(\mu_h + t_h - 1)}{\varepsilon}\}$*;*

[6] With analogous notation as exposed at the beginning of Section 2.1.1, it constitutes an *a posteriori* description of the population, via the sampling mechanism $\mathcal{M}$ rather than an *a priori* measure defined over a σ-algebra on $\mathfrak{X}$.

[7] A usual regularity condition representing the counterpart of a *well-behaved* function request [28]. For formal definitions see [24].

then

$$\mathrm{P}\left(U_{c \div h} \leq \varepsilon\right) \geq 1 - \delta.^{8} \tag{17}$$

Proof (*Sketch*). The claim follows from inverting relations between probabilities and their quantiles in (16) as in Lemmas 2 and 3. Now we are interested in confidence intervals bounded only on the right. Moreover we add to the sentry points also the mislabeled points that by construction fall in $c \div h$ (see [25] for a more detailed discussion on *almost consistent* hypotheses). □

Remark 10. Lemma 9 supplies an upper bound to the size of whatever labeled sample that is necessary to guarantee accuracy ε and confidence $1 - \delta$ to the learning task. Thus, denoting t an upper bound on t_h, we obtain the analogous inequality

$$m \geq \max\left\{\frac{2}{\varepsilon}\log\frac{1}{\delta}, \frac{5.5(\mathsf{D}_{\mathsf{C},\mathsf{H}} + t - 1)}{\varepsilon}\right\}. \tag{18}$$

This equation denotes that the task of learning C through $\mathcal{A}$ has polynomial sample complexity in some characteristic parameter of the problem if $\mathsf{D}_{\mathsf{C},\mathsf{H}}$ and t are polynomial in this parameter.

In the second modality – for instance dealing with a neural network but with our game as well – you simply stretch the hypothesis in a direction that decreases monotonically the symmetric difference between hypothesis and concept (at least in the best case). In this modality you have an essentially on-line learning with no consistency guarantee on the partial sample. Thus the true label you rely on tells you in which examples you win (by computing a correct label) and in which ones you lose (mistaking the label) with no exact connection between their position in $\mathfrak{X}$ and the concept shape. In the companion Game_1 this lack of information is enhanced by the fact that you really do not know the membership function of the example to the concepts but only these comparative win/lose labels. This scenario proves easier to treat in our framework where the concept to be learned is a synthesis of future observation story rather than an *a priori entity* as in the conventional one sketched in Fig. 5(b).

3. Dimensioning the batch size for winning with high probability

Let us fix for the moment the monotone competition to lie in finding a solution to the *knapsack problem* [10] as mentioned in Section 1: given a set of objects, each characterized by a *weight* and a *profit*, find within subsets of them whose total weight does not exceed the *capacity* of a knapsack the one maximizing cumulative profit. As well known, this problem belongs to the NP-hard complexity class [30]. Therefore to avoid operational traps due to exponentially long computations, both competitors adopt polynomial-time approximation schemes (*PTAS*) [31]. In particular we assume for the moment that B implements the Sahni algorithm [11]. In the following we will refer to the following notations:

Definition 11. An instance for the 0–1 knapsack problem is a 4-ple $s = \langle n, W, Z, b\rangle$, where

- $n \in \mathbb{N}$;
- $W = \{w_1, \ldots, w_n\} \in \mathbb{N}^n$;
- $Z = \{z_1, \ldots, z_n\} \in \mathbb{N}^n$;
- $b \in \mathbb{N}$ is such that $w_i \leq b$ for each $i \in \{1, \ldots, n\}$ but $\sum_{i=1}^{n} w_i > b$.

Given a knapsack instance s, an ordered n-ple $\mathbf{x} = \langle x_1, \ldots, x_n\rangle \in \{0, 1\}^n$ is called a *feasible solution* for s if $\sum_{i=1}^{n} w_i x_i \leq b$, and $g(x_1, \ldots, x_n) = \sum_{i=1}^{n} x_i z_i$ is called its *value*. Denoted with $\mathrm{Sol}(s)$ the set of all the feasible solutions for s, a solution of the knapsack problem on s maximizes g over $\mathrm{Sol}(s)$, i.e. is an n-ple

$$\langle x_1^*, \ldots, x_n^*\rangle = \arg \max_{\langle x_1, \ldots, x_n\rangle \in \mathrm{Sol}(s)} g(x_1, \ldots, x_n)$$

where $\arg(f(\mathbf{x})) = \mathbf{x}$.

Denoted with g^* and $\widehat{g}$ the values of an optimal solution and approximate solution, respectively, $\eta = \frac{g^* - \widehat{g}}{g^*}$ is the approximation degree. For given η, Sahni algorithm gets approximate solutions in a time polynomial in the number n of the objects.

8 This bound looks more favorable by a a factor $1/\log\varepsilon$ than the usual one in the literature [28] based on Vapnik–Chervonenkis dimension [27]. This occurs since the two bounds refer to slightly different notions of confidence interval, as mentioned earlier (see Fig. 5).

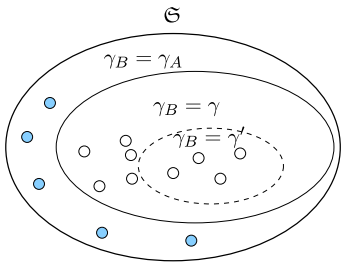

Fig. 7. Partitions of the instance space $\mathfrak{S}$ in Game_1 for different strengths of B, when A has strength γ_A, with $\gamma' < \gamma < \gamma_A$. White balls: instances on which B does not lose when $\gamma_B = \gamma$. Gray balls: instances on which B loses when $\gamma_B = \gamma$.

From an operational perspective, identified the assignments 1 or 0 to x_i with the facts: "the ith object is filled or not in the sack", and denoted γ the ceiling of $1/\eta$, Sahni algorithm enumerates all the feasible solutions composed of at most γ objects and then completes them with a *greedy* routine. This routine [32] fills the knapsack with the unused items sorted in decreasing order of z_i/w_i ratio while $\sum_{i=1}^{n} w_i x_i \leq b$. The approximate solution is one within these solutions which has maximum value.

We assume a sequence T of knapsack instances to be submitted to the two competitors A and B. The latter plays with a strength $\gamma_B = \gamma$: the one who fills the bag more efficiently wins; if both get the same value they tie. For the sake of simplicity, let us assume for a moment that A too uses the same algorithm with unknown strength parameter γ_A. A so defined competition meets the monotonicity requirements fixed in the Introduction: on each s, B can tie, lose or win with respect to A, and the domain h of instances on which B does not lose monotonically increases with γ_B (see Fig. 7).

Definition 12. For a given instance space $\mathfrak{S}$, a γ_B-NL domain is the set of all the instances in $\mathfrak{S}$ on which B does not lose whenever his strength is γ_B. Its complement is denoted γ_B-L domain.

With same notation as in Definition 11, the parameter γ is an integer ranging from 0 to n, where the contour line of the γ_B-NL domain with $\gamma_B = \gamma_A$ exactly contains $\mathfrak{S}$. Actually with $\gamma_B < \gamma_A$ B never wins and with $\gamma_B = \gamma_A$ B always ties. Thus the balls in any γ_B-NL domain in Fig. 7 refer to instances where B always ties with strength γ_B. With a strength greater than γ_A B could either win or tie but his strategy never reaches such a value for γ_B. If we restate the game in terms of a learning task, the goal of B is to learn the γ_A-NL domain, hence a concept c coinciding with $\mathfrak{S}$. But, as B is satisfied when the losing probability is less than a given ε, his hypothesis h is either the entire $\mathfrak{S}$ or a domain inside it like in Fig. 7. Thus the symmetric difference $c \div h$ is reduced to the domain $\bar{h} = \mathfrak{S} - h$. We recognize easily that:

Fact 13. *The detail of the class of symmetric differences between hypotheses and concepts in Game_*1 *is* 1.

Proof. For any domain h among the set of $n+1$ nested γ_B-NL domains corresponding to the possible values of γ as in Fig. 7, consider the set of the complementary γ_B-L domains. A single point representing an instance outside the γ_h-L domain $\bar{h}$ and internal to next $(\gamma_h - 1)$-L domain sentinels the expansion of $\bar{h}$. □

Theorem 14. *In Game_*1 *having a knapsack problem on at most n objects as competition and using the Sahni approximate algorithm, B learns his strength γ_B with accuracy ε and confidence $1 - \delta$ after playing a number m of instances of the problem such that*

$$m \leq \max\left\{\frac{2}{\varepsilon}\log\frac{1}{\delta}, \frac{5.5(n-1)}{\varepsilon}\right\}. \tag{19}$$

Proof. Let us assume that at the beginning of the game $\gamma_B = 0$ (for sake of simplicity), and that the result of our learning task is a strength $\gamma_B = \gamma$. This means that during the game γ losing instances have been submitted to B. As an extreme case let us assume that all γ instances are inside $\bar{h}$ (actually, the true constraint is that the first losing instance s must lie outside the 0-NL domain, the second outside the 1-NL domain, etc.). In this case the right implication in (15) becomes

$$\left(u_{\bar{h}} < \varepsilon\right) \Leftarrow (k_\varepsilon \geq \gamma + 1) \tag{20}$$

which reads: "Given that the game ended with strength $\gamma_B = \gamma$, we acknowledge that $U_{\bar{h}}$ is less than ε because a domain B_{i_ε} of measure ε exists in a nested sequence pivoted around $\bar{h}$ containing from the sample points both the above γ points and a further one outside $\bar{h}$ representing the sentinel of $\bar{h}$". Since having the γ points all inside $\bar{h}$ is less probable than the necessary condition of having the first losing instance s outside the 0-NL domain, the second outside the 1-NL domain, etc., we can assume the probability of the rightmost event in (20) as a lower bound to $\mathrm{P}(U_{\bar{h}} < \varepsilon)$. Moreover, since we do not know what the learnt strength γ_B will be, we further bound from below this probability by substituting γ with its maximum value $n - 1$[9] in the above probability:

$$\mathrm{P}\left(U_{\bar{h}} < \varepsilon\right) \geq \mathrm{P}\left(K_\varepsilon \geq n\right). \tag{21}$$

[9] Actually with $\gamma_B = n$ we are sure that $U_{\bar{h}} = 0$ with no need of a further sentry point.

By inverting this inequality we obtain the claim.[10] □

We extend these results to the whole family of monotone games considered in the definition of Game_1 as follows.

Theorem 15. *In Game_1 having as competition any search problem such that:*

- *a set* Sol(s) *of solutions is defined on each problem instance s,*
- *a total order relation* $\trianglerighteq$ *can be stated between solutions of any such set,*
- *a family of solving algorithms* $\mathcal{A}$ *exists such that for each* $\mathcal{A}$*:*
  - *a strength parameter* γ *exists which in each* Sol(s) *makes* $\mathcal{A}$ *computing one item* $\mathbf{x}$ *for each value of* γ *such that for each pair* $\gamma', \gamma'' : \gamma' > \gamma'' \Rightarrow \mathbf{x}' \trianglerighteq \mathbf{x}''$,
  - γ *ranges in a finite set of values whose number is a polynomial q in the instance length n,*
  - *on maximum* γ *each* $\mathcal{A}$ *computes the same solution,*
- *A and B use an own* $\mathcal{A}$ *each,*
- *B beats A, if* $\mathbf{x}_B \trianglerighteq \mathbf{x}_A$,

on a randomly drawn instance S, B beats A with probability $1 - \varepsilon$ *with a confidence* $1 - \delta$ *after a number of contests polynomial in n,* $1/\varepsilon$, $1/\delta$. *If both competitors use a same* $\mathcal{A}$ *polynomial in n and* γ *(i.e. running on a single s in a time that is polynomial in these parameters), then the learning task has polynomial computational complexity. If such* $\mathcal{A}$ *is polynomial only in n, but A is allowed a polynomial computer time, then the learning task's computational complexity is still polynomial.*

Proof. If we substitute n with $q(n)$ then (21) holds also for the search problems in the claim since the proof of this relation is completely independent of:

- the instance space $\mathfrak{S}$ and the distribution law defined on it;
- the approximation algorithm used by B, provided that it is monotone in a strength parameter;
- the approximation algorithm chosen by A, provided that for maximum γ_B B always either wins or ties.

Consequently, (19) holds as well after the mentioned substitution. Thus we have an upper bound for m that in turn is polynomial on a polynomial in n. Hence the upper bound is definitely polynomial in n. With the same argument of the polynomial collapse we trivially obtain the computational complexity results in the claim. □

Note that the Sahni algorithm is not polynomial in γ. Thus we need the polynomial bound on $\mathcal{A}$ running time to have a feasible game. We remark that other approximation algorithms, such as the one based on the *rounding* technique [33], which are available for the knapsack problem, are polynomial even in γ. With the mentioned algorithms we must however impose a polynomial discretization of the strength parameter that is originally a continuously valued parameter.

4. Guiding a single game

Let us remain for the moment with instances of a knapsack problem and Sahni algorithm for solving them. According to the results in the previous section we can definitely draw a sample (i.e. a set of contests) of size m satisfying (19) to be sure that:

> if we repeat this strategy (play m contests, increase the strength whenever lose) again and again,
> then
> - whatever sample items occur in the single trial,
> - whatever population items occur as their continuation, hence
> - provided a same sampling mechanism $\mathcal{M}$ underlies both,
>
> the asymptotic frequency of hitting the $U_{c \div h} \leq \varepsilon$ target is $\geq 1 - \delta$.

In this section our target is to save contests. Thus we are looking for the minimal number of contests that in the actual game history allow us to assume $U_{c \div h} \leq \varepsilon$ with a given confidence $\geq 1 - \delta$. This means that at run time, after the current contest, B checks (on the basis of statistics on the course of fought contests) whether the above minimum has been reached or not and in the positive case stops the game. Confidence again concerns the asymptotic frequency of hitting the target in many replicas of the adopted decision strategy. We study this strategy on an equivalent game, denoted Game_2, that proves easier to examine:

[10] Note the similarity between this proof and the one of k-CNF learnability [14]. There each updating point gnaws a specific part of $c - h$ in order to make h consistent with it. Here we just push the contour of the NL domain closer to the contour of the whole instance set. We pay the broadness of this strategy in terms of witnesses in (21): notwithstanding the detail of the L domain class is 1, we burn n points.

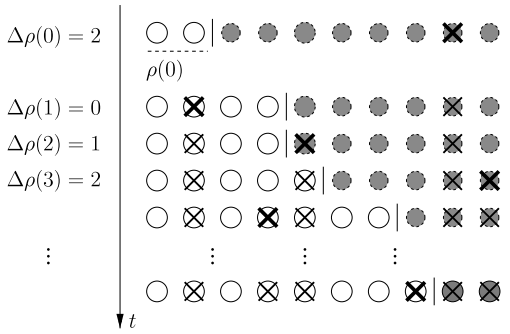

Fig. 8. An instance of Game_2 having $\ell = 10$ and $\rho(0) = 2$. The vertical axis reports the iteration number, each labeled with the corresponding $\Delta\rho(t)$. Marked balls are denoted by a cross, while bold crosses denote the balls drawn at each iteration. The cursor position is understood before the ball extraction and eventual updating of it.

Game_2. *Consider the atomic partition of the unitary probability mass in a set of balls as in the first row of Fig. 8; let ℓ be the number of balls and $\frac{1}{\ell}$ the probability of drawing a ball. Starting the game, the first $\rho(0)$ balls are white while the remaining ones are gray, and a cursor separates the two sets of balls. At each run t we draw a ball uniformly and replace it after having marked it: if the ball is white we do nothing more, if it is gray we move the cursor $\Delta\rho(t)$ balls right, with $\Delta\rho(t) \geq 0$, changing color to the shifted balls (i.e. the new balls to the left of the cursor become white). $\Delta\rho(t)$ is not fixed a priori and can change from iteration to iteration.*

There is a strong relation between Game_1 and Game_2: indeed, for a fixed value γ_A of A strength and at a current γ_B strength, we can think of the balls as probability masses associated to knapsack instances, where their number in the association determines the probability of the instances. We state nothing about its distribution. We simply color the balls according to their corresponding contest attitude, white if indicative of a B victory or tie and gray otherwise; we assign contiguous balls to a same instance so that proper shifts of the cursor move an instance completely from the white to gray section. Drawing a white ball corresponds to an actual round of a non-losing instance for B. In this case both games do nothing (apart from the ball marking for statistical reasons). Drawing a gray ball corresponds to losing a contest. In this case, the cursor shift of $\Delta\rho(t)$ units right figures the unitary increment of γ_B. Both moves have the effect of decreasing (by an unknown quantity[11]) the probability of drawing losing instances in the game's run.

Claim 16. *Game_1 and Game_2 are equivalent.*

Proof. For a given strength γ_A because of the uniform monotonicity of the competition, we can partition the instance space of Game_1 through the γ_B-NL domains introduced before (see Fig. 7). As the domains are nested we can order the instances through their contour lines. Moreover, since both $\mathfrak{S}$ and γ_B range are finite and discrete, we can discretize the unitary probability mass in a number ℓ of equally probable atoms so that each ring between contour lines in Fig. 9 contains exactly an integer number of these atoms. Finally, we project the atoms on the real line (right side of Fig. 9) such that the order relation introduced by the contour line is preserved. In this way:

- the first $\rho(0)$ atoms are the projection of the initial γ_0-NL domain, where γ_0 is the starting strength of B;
- at each unitary increment of γ_B the enlargement of the NL domain maps into a cursor jump of $\Delta\rho$ atoms left-to-right along the line;

and the equivalence of the two games follows. □

Thus we can represent a history of Game_1 through a history of Game_2, gaining in representation simplicity.

Claim 17. *Let us set at ℓ (unknown and as large as necessary) the number of balls (hence the probability discretization grain) and denote with K the random variable representing the number of updates of the cursor position. Correspondingly we denote with I_K the set of balls to the right of the cursor (the gray balls) after K updates and with U_K its probability measure. In particular, with K_ε we refer to the number of updates when the measure of the gray balls is ε. Given the peculiarity of our game, to focus on a particular extraction of balls we need to identify not only the balls drawn but also their drawing sequence. However, for any such specification:*

$$(u_k \leq \varepsilon) \Leftarrow (k_\varepsilon \leq k). \tag{22}$$

Proof. The statement simply follows from the fact that a left-to-right shift of the cursor expands (or at least does not narrow) the white section. □

[11] Unawareness of ℓ, $\rho(0)$ and $\Delta\rho(t)$ differentiates this drawing mechanism from Polya's urn model [34].

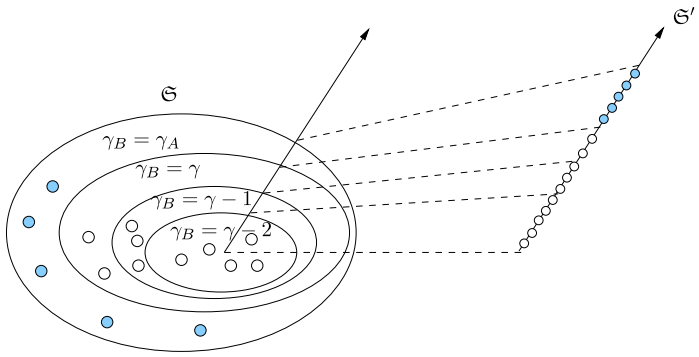

Fig. 9. Strength contour lines partitioning the instance space (left diagram) and their corresponding probability masses projected on the real line (right graph).

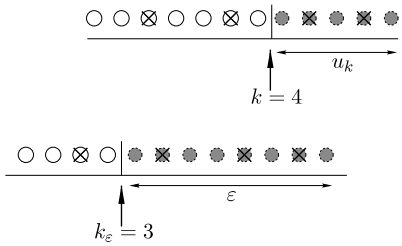

Fig. 10. The actual process giving k updates and the probability measure u_k, and a hypothetical one giving ε for a suitable drawing sequence of the same balls set (for instance in the former the sequence is left to right and *vice versa* in the latter) and cursor shifting strategy.

Claim 18. *Let us denote by $\widetilde{k}$ the number of marked balls in I_k, i.e. the marked balls that remain on the right of the cursor after k updatings (this means that they were gray when sampled and remain gray at the current step of the game). Then* for a same set of m balls drawn *resulting in k updatings in the game specified by the pair $\langle \rho(0), \Delta\rho(t) \rangle$ and corresponding to a given drawing sequence, consider any drawing sequence and any game whose $\rho'(0)$ and $\Delta\rho'(t)$ move the cursor so that at $t = m$ it reaches a position that leaves on the right a probability measure $\varepsilon \in \{0, \ldots, \frac{i}{\ell}, \ldots, 1\}$ (see Fig. 10), and denote with $\widetilde{k}_\varepsilon$ the analogous of $\widetilde{k}$ in the new game. For any set of sampled balls*

$$(\widetilde{k}_\varepsilon \geq \widetilde{k}) \Leftarrow (u_k \leq \varepsilon). \tag{23}$$

Proof. At the end of sampling, the intersection between the sets of gray balls in the two games (i.e. the set to the right of the rightmost cursor) includes the same set of marked balls in both games. Hence, the fewer these balls, the fewer gray balls in general. Therefore

$$(u_k > \varepsilon) \Leftarrow (\widetilde{k}_\varepsilon < \widetilde{k}). \tag{24}$$

The claim follows after considering the complementary events. □

Theorem 19. *With reference to Game_2 and variables K, U_K and $\widetilde{K}$ as in Claims 17 and 18, denoting by*

- *m the number of steps of our game,*
- *Θ_ε the Binomial variable of parameters m and ε, and Ψ_ε the Binomial variable of parameters k and ε,*

we have

$$\mathrm{P}(\Theta_\varepsilon \geq \widetilde{k}) \geq \mathrm{P}(U_k \leq \varepsilon) \tag{25}$$
$$\mathrm{P}(U_k \leq \varepsilon) \geq \mathrm{P}(\Psi_\varepsilon \geq \widetilde{k} + 1). \tag{26}$$

Proof. (1) From (23) $\mathrm{P}(\widetilde{K}_\varepsilon \geq \widetilde{k}) \geq \mathrm{P}(U_k \leq \varepsilon)$. We can figure the ordered balls in a strip in Fig. 8 as the projection of a discretized version of the U bars in Fig. 1, where the marked balls represent the extremes of these bars. The cursor shift between the strips could correspond to the movement of the threshold line. The true difference between the two inference schemes lies in the fact that the cursor motion is ruled by the sample itself according to our game strategy. This means that the drawings of gray balls from a domain whose measure is ε are not independent; rather, the probability of such extraction conditioned by a previous extraction of this kind is less than the unconditioned probability of the same event. Thus $\widetilde{K}_\varepsilon$ is approximated with the Binomial variable of parameters m and ε, expressly to obtain an upper bound. This inequality extends to the ε's not belonging to $\{0, \ldots, \frac{i}{\ell}, \ldots, 1\}$. Indeed, denoting with ε' the highest value in this set less than ε, we have $(U_k \leq \varepsilon) \Leftrightarrow (U_k \leq \varepsilon')$, and $\mathrm{P}(U_k \leq \varepsilon) \leq \mathrm{P}(\Theta_{\varepsilon'} \geq \widetilde{k}) \leq \mathrm{P}(\Theta_\varepsilon \geq \widetilde{k})$.

(2) Using relation (22) to fix a lower bound to the U_k c.d.f. is hard because of the mentioned probabilistic conditioning between the updating balls. Thus we use another strategy. Let us denote *eligible set* σ_i the set of balls from which the ith updating ball is drawn and Z_i its index in the whole ball set. Consider the initial set of gray balls and, included in it, the set of balls to the right of the cursor at the kth update, i.e. σ_1 and σ_{k+1}, respectively. Here σ_{k+1} represents the set from which we can extract a further updating ball if we continue the sampling.

If we adopt the simplifying hypothesis that the k updating balls come from k independent extractions from σ_1 we can consider the ordered sequence $Z_{(1)}, \ldots, Z_{(k)}$ of their indices, called *rank order statistics* elsewhere [23], and the set of random segments $\{(Z_{(1)}, Z_{(2)}), (Z_{(2)}, Z_{(3)}), \ldots, (Z_{(k)}, \infty)\}$, called *statistically equivalent blocks* since the measure of each block follows the same Beta distribution law of parameters 1 and k [20]. In this model σ_{k+1} is included in the union of the last $\widetilde{k}+1$ blocks. Indeed it contains the last $\widetilde{k}$ blocks but is contained in the union of the former ones with the block subsequent to the left to $Z_{(\widetilde{k})}$, since we know that the left extreme of this block changed color at one of the k updatings. Moreover, as discussed in point (1) in the actual process these blocks have a possibly smaller measure due to its conditioning by the strength updating strategy.

In sum, the probability measure of $\widetilde{k}+1$ blocks follows a Beta random variable of parameters $\widetilde{k}+1$ and $k-\widetilde{k}$. The distribution law of U_k is minorized by the above Beta distribution for two reasons:

(a) in regard to the eligible sets, U_k is the measure of the $(k+1)$th one. By definition the unconditioned measure of this set is less than any conditioned one (provided that the conditioning event includes σ_{k+1}). Thus for I_k as in Claim 17 and a ball D randomly drawn

$$\mathrm{P}(U_k \leq \varepsilon) \geq \mathrm{P}(\mathrm{P}(D \in I_k \mid D \in I_0) \leq \varepsilon) \tag{27}$$

(b) as for Beta distribution, in the first place, the probability that $\widetilde{k}+1$ statistically equivalent blocks have a probability measure $\leq \varepsilon$ (i.e. the c.d.f. of these blocks) equals the probability of drawing at least $\widetilde{k}+1$ gray balls in a region of measure ε (the rightmost term in (26)). In own turn we proved that the measure of the actual eligible set σ_{k+1} is smaller or equal to the measure of the $\widetilde{k}+1$ blocks, and an opposite relation holds between the probabilities for their measures to be less than a given value. Therefore the c.d.f. of the measure of σ_{k+1} conditioned by its inclusion in σ_1 is greater than the c.d.f. of the Beta variable mentioned above:

$$\mathrm{P}(\mathrm{P}(D \in I_k \mid D \in I_0) \leq \varepsilon) \geq \mathrm{P}(\Psi_\varepsilon \geq \widetilde{k}+1). \tag{28}$$

Thus chaining (27) and (28) we get (26). □

Corollary 20. *At any step m of either Game_1 having a competition as in Theorem 14 or Game_2 equivalent to the former such that the number of strength/ cursor updates is k and the number of remaining defeats/gray marked balls is $\widetilde{k}$, the tail symmetric confidence interval of level less or equal to δ for the probability of a B defeat is (l_i, l_s) where l_i is $\delta/2$ quantile of the Beta distribution of parameters $\widetilde{k}$ and $m-\widetilde{k}+1$, and l_s is the analogous $1-\delta/2$ quantile for parameters $\widetilde{k}+1$ and $k-\widetilde{k}$.*

Proof. Simply by considering the analytical form of the distribution laws of Θ_ε and Ψ_ε we obtain

$$I_\varepsilon(\widetilde{k}, m-\widetilde{k}+1) \geq \mathrm{P}(U_k \leq \varepsilon) = F_{U_k}(\varepsilon) \geq I_\varepsilon(\widetilde{k}+1, k-\widetilde{k}). \tag{29}$$

Then we compute the confidence interval extremes in the same way as in Lemmas 2 and 3. □

Remark 21. An extension analogous of Theorem 14 may be enunciated for Theorem 19 to cover monotone games as in Theorem 15 in general.

We performed some direct simulations of Game_1 using the format adopted earlier. Here the unitary uniform variables are used as a source of randomness for extracting the sequence with which instances s of a knapsack problem are picked from a set $\mathfrak{S}$ wisely manufactured. According to Corollary 20, the confidence interval upper bounds move from those relative to the pure Bernoulli case to include, with a given confidence the population parameters. We properly stretched the instances' distribution in order to exploit this facility through the 300 curves in Fig. 11(a). They refer to initial steps of games each having an own set $\mathfrak{S}$ of 45 knapsack instances made up of 12 objects with variously unbalanced weights, profits and capacity. Each game runs on 10 instances picked at random. A fixes γ_A randomly within the set $\{1, \ldots, 10\}$, B plays out his strategy by increasing γ_B by 1 on each defeat with respect to A. To gain an overall appreciation of the coverage of the confidence intervals by the trajectories of U_k, on each move we normalized the extremes of the 90% confidence interval and the actual u_k specification. Namely, on each m we assumed the mean (over the 300 trajectories) of confidence interval centers as the baseline intercept and the mean of the interval half widths as normalization factor.[12] Hence the contour of the confidence regions collapse to the two black bold lines. To fill the confidence region we considered three families of curves coming from different instances distribution laws and initial B strengths. We colored each family with a different gray tune and plotted the curves sequentially from form darker to lighter ones. Due to their positioning within the picture the former are mostly hidden by the other two families.

[12] In formulas we map u_k into $\breve{u}_k = \bar{u}_k + \frac{\bar{l}_s - \bar{l}_i}{\bar{l}_s - \bar{l}_i} u_k$ where $\bar{a}$ is the sample mean of a for fixed m.

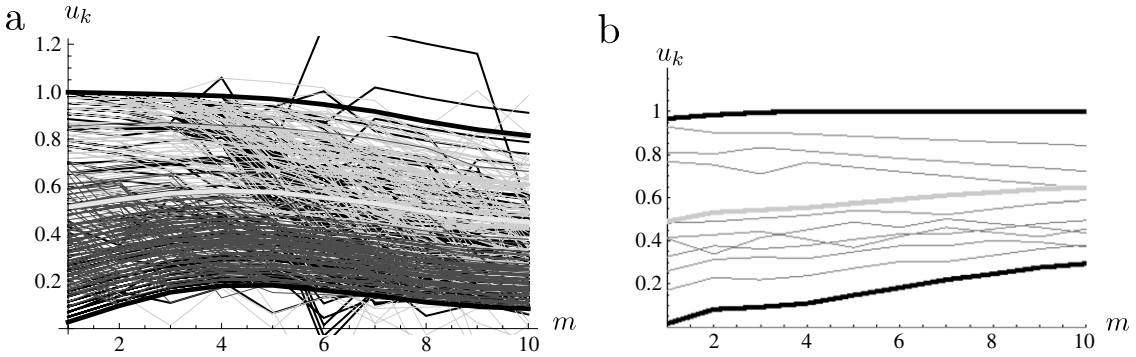

Fig. 11. Game histories disputed on 10 knapsack contests with $n = 12$. Horizontal axis: time step m. On the vertical axis: (i) thin lines: u_k, different gray scales correspond to different instance distributions; (ii) bold black lines: 0.9 confidence interval for u_k computed through (29); (iii) bold gray line: baseline. (a) overall pictures of the normalized tracks; (b) dummy training histories with constant strengths.

In greater detail, the 100 black curves refer to *equally distributed* instances of a game starting with $\gamma_B = 0$. The trend of u_k losing probabilities goes from high values (whose normalized version may be greater than 1 or less than 0 because of rescaling) at the start of the game to low values at its end denoting a good efficiency of the learning strategies. Things get worse if the instance distribution is heaped toward the two extremes constituted by *easy* instances (solvable by B with the same efficiency as A though using a smaller strength) and *hard* instances (where B loses as soon as he uses less strength than A). In these cases γ_B increments often do not affect the losing probability giving rise to the almost horizontal trend of the gray curves in the picture: specifically, light gray curves start with $\gamma_B = 2$, meaning a high initial losing probability; dark gray ones start with $\gamma_B = 5$, yielding the opposite effect. The curves cover nearly the entire confidence region with a few trespassing over its borders. The reader may expect a proper pairing of further stretched initial strengths with suitable instance distributions to complete the coverage, apart from a small oversizing of the confidence region, specially on the top, due to the inequalities in (29). Evidence of this expectation is shown in Fig. 11(b) where, having fixed $\gamma_A = 10$, we report a dummy training history for each initial B strength between 1 and 9 raising from uniformly distributed instances in $\mathfrak{S}$ with 0 increment on every defeat.[13] Analogous curves coincide with lower and upper bounds when $\gamma_B = 10$ or when on no instance B ties with $\gamma_B = 0$, respectively.

5. Overcoming indeterminacy and companion overtraining phenomena

5.1. Appreciating the training strategy efficiency

Theorem 19 appears somehow to be an extension of the *Perceptron Theorem* [35] concealing the same kind of indefiniteness. As long as your learning algorithm gains monotone improvement of the cost function you get an *adequate* solution with a *number k of updates* that scales with the characteristic dimension n of the problem. Apart from the vagueness of some terms that we will discuss shortly, the main point is that you must work with consistent statistics (identified with the sample cost function to be minimized) in order to realize the monotonicity of the population cost function with the sample size (i.e. to deal with a Lyapunov function on this variable). This gives a rationale to the search for the best error function (such as MSE [36], Kullback distance [37], posterior probability [38], etc.), and suggests a pre-eminence of the shape of the function in respect to the numerical tricks to minimize it. Rather, the latter concern the time complexity, in close analogy to what mentioned in concern of PTAS algorithms. Then we must define *adequacy*. Our template game suggests interpreting it as a feature characterizing solutions that are good enough provided that no real computing device, nature included, may rely on unlimited computational power (companion of the notion of equivalent hyperplanes in the Perceptron Theorem).

As a new theoretical facility, Corollary 20 deals with the efficiency of the training strategy as well. Because the solutions of the quantiles l_i and l_s are a function of the two parameters k and $\widetilde{k}$ for a given length m of the game history, we draw the nice three-dimensional graph (which we will call *sails' graph*) in Fig. 12 to represent them. The main feature of this picture is the distinctive presence of: (i) a squeezed region where we would like to locate our confidence interval that would prove to be tight around low U_k values and (ii) a broad region with no meaningful information about U_k. One great quality of the picture is that between the upper and lower surfaces we can frame trajectories like those in Fig. 11 with a good approximation, getting the game timing as fourth implicit coordinate. Actually, the trajectories mentioned refer to the first 10 steps of the game, while Fig. 12 refers to statistics that should be collected after exactly 10 steps. However, from (29) we see that this discrepancy does not affect the interval upper bounds, while lower bounds with $m = 10$ are lower than the analogous bounds computed at run time with m exactly equal to the number of currently fought contests.

13 Not violating, however, the specifications of a monotone game.

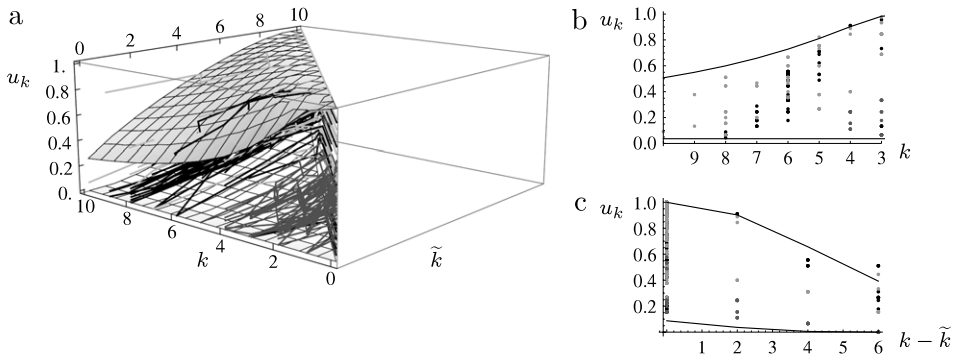

Fig. 12. 0.9 confidence intervals for losing probability. (a) Course of the lower and upper bounds (u_k axis) with statistics k and $\widetilde{k}$ ranging between 1 and 10. The lines between the two surfaces represent the game stories of Fig. 11(a); $m = 10$. (b)–(c) Sections of the picture (a) with the plane $\widetilde{k} = 2$ and $k + \widetilde{k} = 6$, respectively.

From a statistical perspective the so filled sails in Fig. 12(a) and their sections in Fig. 12(b) and (c) show that the large intervals are really spanned by the defeat probability. In addition, we may analyze the individual training histories in terms of efficiency of the learning strategy employed as follows:

(1) a good training occurs when $\widetilde{k}$ goes to zero with increasing k, thus pushing the training trajectory toward the $\langle k = \max, \widetilde{k} = 0\rangle$ corner of the graph in Fig. 12(a). This denotes that the strength updates tangibly reduce the L domain,[14] hence the number of sampled instances remaining in it. The timing of the updates (i.e. their course with the number of processed instances) depends on the current measure of this domain in a distribution-free way;
(2) while proving common at the beginning of the histories, a persistence of $\widetilde{k}$ close to k along a track denotes an almost stationary L domain. This brings the learning problem close to a simple estimate of the Bernoulli parameter measuring the probability of the domain. The wider confidence interval we obtain crossing the sails with the vertical line $\widetilde{k} \approx k = c$ for any constant c vis à vis the analogous interval coming from Fig. 2 with $\phi = \widetilde{k}/m$, depends on the updates' timing. From (29) the upper bound in Fig. 12 is unable to consider at which moment of the sampling story the pair $\langle k, \widetilde{k}\rangle$ has been reached, hence whether after the whole m-sized sample has been drawn or not. It conservatively manages a dummy situation where just k instances have been *independently* drawn, $\widetilde{k}$ of which falling in the losing domain.

The scarce efficiency of the second category of training stories recalls a sort of indeterminacy situations figuring that in certain games *we cannot obtain simultaneously for B both a low defeat probability measure and a low confidence interval*. We go thoroughly through this phenomenon as a template of learning difficulties and key point for stressing learning efficiency, as well. From a logical perspective all steams from the implication direction in (22). Due to the learning dynamics we expect a greater k, hence a longer list of processed defeats, exactly when the current defeat probability is lower (defeat controllability perspective). In contrast, as soon as this probability decreases you expect to meet a shorter sequence of losing instances (defeat observability perspective). The balance between the two arguments determines the wide spectrum of equilibrium points. We numerically sampled them with trajectories of Fig. 11 that may be suitably reread in these terms. You may start with high losing probability u_0 and draw a few instances, consequently observing a small k and $\widetilde{k}$. If the strength increments are not effective you stay around $\langle k, \widetilde{k}, u_0\rangle$ point. Otherwise you lower the losing probability according to the increase in efficacy. Conversely, you may start with low probability of loss and not observe defeats. Continuing the training story, i.e. incrementing the training set, in the first case you move close to the $k = \widetilde{k}$ diagonal. In the second case you remain in the area of the axes origin. Intermediate trajectories deviate from the diagonal with the efficacy of the strength. Starting with still lower initial probabilities you have similar behavior with lower vertical coordinates and a slower timing of the moves with the instances' processing.

In any case the losing probabilities converge to the one connected with the final B strength, and both $\frac{\widetilde{k}}{m}$ and $\frac{k}{m}$ too. All these quantities go to 0 for unconstrained γ_B. But our learning strategy may decide – either voluntarily, since requested confidence has been reached, or involuntarily, just because of our unawareness – giving either zero or ineffective increments from a certain γ_B on (see Footnote 13). Correspondingly, u_k tracks a horizontal trajectory. In conclusion U_k really spans any part of the confidence region with substantially blind training trajectories. A better relation than (29) of u_k with m could lower the confidence region upper bound, but finding this relation may be a hard task in the game problem, that proves unfeasible in subsymbolic learning, as will be explained in a moment. On the contrary, we may set up a simple cognitive artifice to better locate points in the graphs, and thereby obtain an indeterminacy lowering. From point (1) we indeed appreciate that we get better estimate when the difference between k and $\widetilde{k}$ is meaningful. We obtain it by playing two games simultaneously: an *effective game*, where Bob uses the current strength to compute $\widetilde{k}$; and a *dummy game*, featuring a strength increment equal to zero, which lets him accumulate k. These statistics are the same as in a *virtual game* which, starting from 0, moves

14 See Definition 12 in whose respect we dropped the strength prefix γ_B- for conciseness sake.

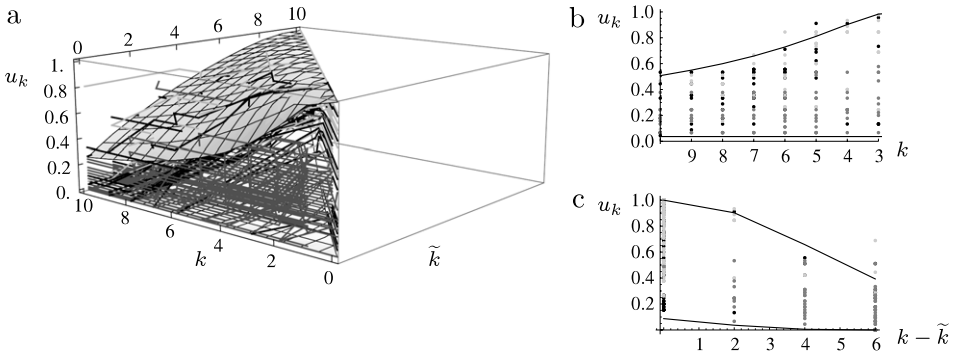

Fig. 13. 0.9 confidence regions for a virtual game. Same notations as in Fig. 12.

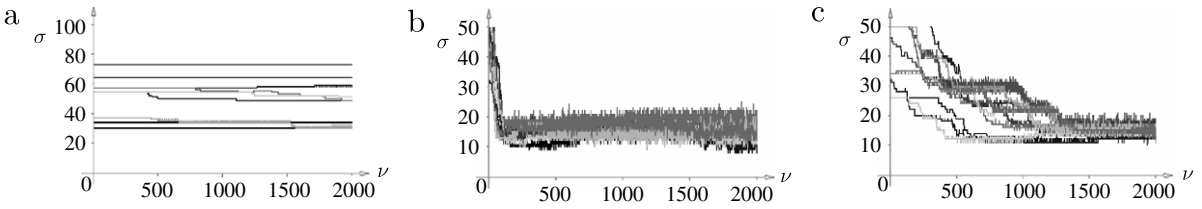

Fig. 14. Courses of the cost function σ with the number ν of training iterations for a subsymbolic learning problem. Pictures (a), (b) and (c) refer to different settings of the free parameters.

from this value and reaches Bob's current strength exactly when we compute the confidence interval. Fig. 13(a) shows that this strategy pushes the great part of the trajectories toward the $\langle k = \max, \widetilde{k} = 0\rangle$ corner. Consequently, by contrasting the two sections in Fig. 13(b) and (c) with those in Fig. 12(b) and (c), we see a migration of line intersections from broader to narrower confidence intervals. Thus we obtain a better appreciation of the confidence intervals for the same learning process if we exploit more information on them and this information is available.

5.2. Subsymbolic learning

The final aim of this paper is to use sails in Fig. 12 as a state diagram of any subsymbolic learning process in order to draw a few well-founded directions for efficiently driving it. To render still more blind the trajectories of the training stories here we also lack: (i) the knowledge of n as an actual complexity measure of the learning task, and (ii) a clear connection between the statistics k and $\widetilde{k}$ and the distribution law of the goal function u_k.

The typical tool for analyzing a training session is the course of the cost function as reported in Fig. 14. We will use the virtual game strategy for relating features of these graphs to the features of the sails' graph. In this perspective, $\widetilde{k}$ plays the role of the current cost function statistic σ, for instance the sum of square errors, and k the one of its initial value σ_0; u_k plays the goal function τ (for instance the mean square error) that we want to minimize in our learning strategy.

In particular, when we learn Boolean functions, under the considered monotonicity assumption of the training algorithm, k and $\widetilde{k}$ maintain the same semantic (number of mislabeled points with the initial and current hypotheses) and u_k coincides with $u_{c\div h}$ introduced in Section 2.2.2. Thus we have the following corollary.

Corollary 22. *Given an algorithm $\mathcal{A}$ for incrementally learning a Boolean function c from examples, if the measure $u_{c\div h}$ of the symmetric difference of c with the function h computed by $\mathcal{A}$ is a non-increasing function of the number of processed examples, then the numbers k and $\widetilde{k}$ of mislabeled examples at respectively the beginning and the current status of training are pivots for a twisting argument on $U_{c\div h}$.*

Proof. We may map the learning procedure steps into those of the virtual game considered at the end of Section 5.1. Thanks to Theorem 2, the sole difference with respect to the probabilistic framework of Game_1 is that here we lack the dimensionality n of the problem. It depends on the training algorithm (see Footnote 10), and does not necessarily coincide with the dimensionality of the examples. However, in the proof of Theorem 2 we have realized that, for whatever n, the two statistics pivot twisting arguments about $U_{c\div h}$. □

Referring to non-Boolean functions we focus on the cost functions σ_0 and σ at respectively initial and current learning time. Time elapses for two reasons, i.e. along two coordinate axes: (i) we process a certain number of examples, and (ii) we do it thoroughly by iterating a training procedure. In this respect k, i.e. the number of updates or correspondingly the σ evolution, represents the true *internal time* of the process.[15] The monotonicity assumption of the cost function with the

15 This is why we denote with the symbol u_k the defeat probability, being $\widetilde{k}$ a special value of k.

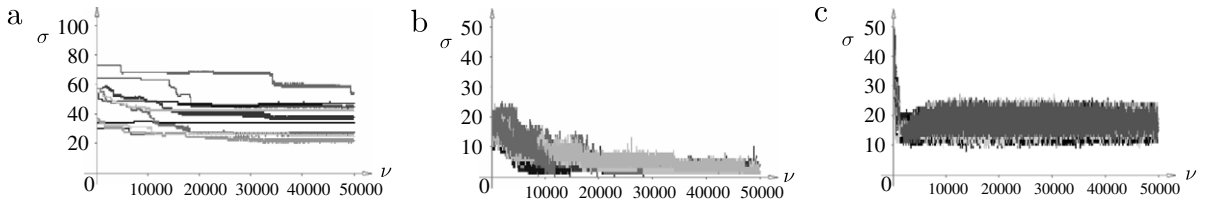

Fig. 15. Same graphs as in Fig. 14 but referred to longer training histories.

number of examples allows us to deal separately with the two coordinate axes. In the game problem we consider explicitly the first coordinate m shading the other in the learning efficiency. For a given training strategy, looking at the learning tracks behavior we may say that a training session is *de facto* finished when the $\langle \frac{k}{m}, \frac{\widetilde{k}}{m} \rangle$ trajectory gets stuck in an equilibrium point, as mentioned in Section 5.1. Running along the first time coordinate, at that point the testing session starts since the new instances refer to an unchanging L domain, *ergo* with no burning of new witness points or, equivalently, having $(\mathsf{D}_{\mathsf{C},\mathsf{H}})_h = 1$ in (16) [25]. This looks for a dummy on-line learning strategy where no partition is made in advance on the available examples between training and testing set. We train the network progressively on the examples being satisfied when no significant changes occur on σ. Rather, we could actually train the network on the whole training set having for obvious that the best use of an example is for training a network rather than for testing whether the training set was sufficiently large or not.

The principal use of a test set is however to understand the efficiency of the training strategy, a feature referring to the second time coordinate that both determines in own turn the above equilibrium point, and is hardly appreciable in advance. Nevertheless we will still try to get some directions from Fig. 14, hence on the basis of sole training session statistics. In very broad terms we may say that a training loop has reached an efficient training level when it has pushed the training trajectories in the left corner, i.e. with $\widetilde{k}$ far less than k. But in our subsymbolic strategy we are not able to appreciate how far $\widetilde{k}$ is from k in the Boolean case (lack of n), and which metering to use at all for the statistics in the general case. Hence we may use the statistics themselves as faithful estimators of the goal function τ and appreciate their variance at any stage of the training process for locating this stage within the sails' graph. Namely, for a given number of training iterations we consider a certain number of runs coming from different random initializations of the parameters to be learnt, and appreciate the variance of the cost function[16] (or equivalently its square root, i.e. its standard deviation (std)), somehow in line with the bootstrap philosophy [39]. We see from the sails' diagram that a favorable feature is represented by an intermediate value of std. Indeed at one extreme (very high variance) we are in the region with $k, \widetilde{k}$ both low, denoting a slow internal time where any value of u_k or the equivalent τ is possible. If this feature occurs after many actual training iterations we may understand that there is no way of improving the process. We meet the other extreme (very low variance) when the virtual time has elapsed. This may denote a satisfactory situation if we are in the $\langle k_{\text{high}}, \widetilde{k}_{\text{low}} \rangle$ corner, saying that the training has been successfully accomplished; or it may indicate a bad situation as we are in the $\langle k_{\text{high}}, \widetilde{k}_{\text{high}} \rangle$ corner, hence did not learn anything and do not expect it in the future. As for the scale through which to appreciate std values we observe that in the expected value of lower bound distribution (29) – which determines the upper bound to the population error – k and $\widetilde{k}$ linearly scale. Indeed the expected value of $I_\varepsilon(\widetilde{k}+1, k-\widetilde{k})$ is $\frac{\widetilde{k}+1}{k+1}$. Hence we may assume the initial value of the cost function std as our metering unit. Putting together these considerations we may fix the following rule for stopping a training:

Rule 1. Starting from a set of m examples to learn a function f by minimizing a cost function σ as good pivot for minimizing a goal function τ:

use all the examples as a training set;
if σ goes satisfactorily fast to 0 with training iterations
then you are OK,
else having reached at a given iteration t a suspicious almost stationary trend of σ,
repeat the training algorithm many times starting from a different random initialization of the parameters to be learnt and stopping each time at same iteration t,
if the variance of the last few values of σ merged between the iterations is moderately large,
then continue the training,
else stop the training.

Far from pretending to provide an exhaustive check of this rule – a task as wide as the province of the rule – Fig. 15 represents the continuation of the training stories in Fig. 14, giving some evidential support in favor of it. The graphs refer to the solution through a three-layer perceptron of the MIRRORING problem: reproduce on a set of visible neurons the same state vector of another set of visible neurons under the constraint that no direct connection exists between the two sets. In particular we considered training sets made of orthogonal Boolean state vectors of ten bits. At iteration 2000 only the graph in Fig. 15(b) suggests continuing the training. And in fact it gets better σ values later on. The graph in Fig. 15(a) denotes a definitely higher

16 Theorem 15 suggests that we do not need a huge number of trajectories for appreciating the variance.

variance due to an initialization of the network weights very far from zero. Hence the connections saturate their adapting capability in a few cycles. The graph in Fig. 15(c) suffers from an excessively small number of hidden nodes (actually only one against the three nodes of the network trained in Fig. 15(b)). The networks are well trained (the equivalent k is large) but cannot avoid a large percentage of mistakes. Note that the actual difference between the std in this case and the first one is even higher than it appears because of scale change. We must take into account that in graph (c) we start from an initial high std and decrease it with cycles. The opposite occurs in graph (b).

6. Conclusions

Framing a learning task into stochastic game paradigm lets us stress controllability problems of the learning process in very severe boundary conditions. We focus on the poor knowledge available in subsymbolic learning algorithms, such as widespread back-propagation. In this framework we obtained: (i) exact results concerning the problem of learning the strength parameter of a monotone game, (ii) analogous results for learning a Boolean concept, and (iii) a principled rule for deciding when to stop a training session.

The main point is that subsymbolic incremental learning algorithms cannot ensure the consistency of the hypotheses. Rather, they aim for a monotone decrease in the probability of incorrect computing if we base ourselves on these hypotheses. Assuming that this target has been achieved, we use the typical monotone argumentation of Algorithmic Inference for computing the distribution law of the above probability on instances' populations that are compatible with the actual set of examples. This compatibility pivots on two statistics estimating the incremental benefit of the learning strategy with the processed examples.

As a conclusion, in this paper we give a few directions for getting a reasonable training rule supported by a bulk of theoretical arguments and a few numerical experiments. Their value lies in the fact that they are independent of the actual learning task. Further work in this direction will concern:

- rereading ensemble learning techniques [40] in terms of games between teams, and
- exploiting knowledge on strength increase effects for specific learning tasks, for instance in terms of smart statistics capturing the computational structure of a task, as the joint $\langle K, \widetilde{K} \rangle$ considered in this paper do at an early stage.

References

[1] J. Nash, Non-cooperative games, Annals of Mathematics 54 (1951) 286–295.
[2] E. Kalai, E. Lehrer, Rational learning leads to Nash equilibrium, Econometrica 61 (1993) 1019–1045.
[3] D. Foster, R. Vohra, Regret in the on-line decision problem, Games and Economic Behavior 21 (1997) 40–55.
[4] A. Roth, I. Erev, Learning in extensive form games: Expermimental data and simple dynamic models in the intermediate term, Games and Economic Behavior 8 (1995) 164–212.
[5] T. Borgers, R. Sarin, Learning through reiforcement and replicator dynamics, Mimeo, 1997.
[6] D. Blackwell, M.A. Girshick, Theory of Games and Statistical Decisions, Dover Publications, Inc., New York, 1979.
[7] T. Sandholm, R. Crites, Multiagent reinforcement learning in the iterated prisoners' dilemma, Biosystems 37 (1995) 147–166 (special issue on the Prisoners' Dilemma).
[8] P. Auer, N. Cesa Bianchi, Y. Freund, R. Schapire, Gambling in a rigged casino: The adversarial multi-armed bandit problem, in: Proceedings of the 36th Annual Symposium on Foundations of Computer Science, ACM Press, 1995, pp. 322–331.
[9] D. Gale, Monotone games with positive spillovers, Tech. Rep., Dept. of Economics, New York University, 2000. Available on-line: http://netec.mcc.ac.uk/WoPEc/data/Papers/fthstarer9834.html.
[10] S. Martello, P. Toth, The 0–1 knapsack problem, in: Combinatorial Optimization, Wiley, 1979, pp. 237–279.
[11] S. Sahni, Approximate algorithms for the 0/1 knapsack problem, Journal of the Association of Computing Machinery 22 (1) (1975) 115–124.
[12] A.L. Blum, R.L. Rivest, Training a 3-node neural network is NP-complete, Neural Networks 5 (1) (1992) 117–127.
[13] S.S. Wilks, Mathematical Statistics, in: Wiley Publications in Statistics, John Wiley, New York, 1962.
[14] L.G. Valiant, A theory of the learnable, Communications of the ACM 11 (27) (1984) 1134–1142.
[15] B. Apolloni, D. Malchiodi, S. Gaito, Algorithmic Inference in Machine Learning, in: Advanced Knowledge Intenational, Magill, Adelaide, 2003.
[16] B. Apolloni, A. Esposito, D. Malchiodi, C. Orovas, G. Palmas, J. Taylor, A general framework for learning rules from data, IEEE Transactions on Neural Networks 15 (6) (2004) 1333–1349.
[17] B. Apolloni, D. Malchiodi, C. Orovas, G. Palmas, From synapses to rules, Cognitive System Research 3 (2002) 167–201.
[18] P.S. Laplace, Philosophical Essay on Probability, Springer-Verlag, 1995, originally published 1825.
[19] M.A. Fisher, The fiducial argument in statistical inference, Annals of Eugenics 6 (1935) 391–398.
[20] J. Tukey, Nonparametric estimation II. Statistically equivalent blocks and multivariate tolerance regions. The continuous case, Annals of Mathematical Statistics 18 (1947) 529–539.
[21] B. De Finetti, Theory of Probability. Vol. 2: A Critical Introductory Treatment, John Wiley & Sons, New York, 1975.
[22] V.K. Rohatgi, An Introduction to Probablity Theory and Mathematical Statistics, in: Wiley Series in Probability and Mathematical Statistcs, John Wiley & Sons, New York, 1976.
[23] D.A.S. Fraser, Nonparametric Methods in Statistics, John Wiley, New York, 1965.
[24] B. Apolloni, S. Chiaravalli, PAC learning of concept classes through the boundaries of their items, Theoretical Computer Science 172 (1997) 91–120.
[25] B. Apolloni, D. Malchiodi, Gaining degrees of freedom in subsymbolic learning, Theoretical Computer Science 255 (2001) 295–321.
[26] I. Wegener, The Complexity of Boolean Functions, Teubner and Wiley, 1987.
[27] V. Vapnik, Estimation of Dependencies Based on Empirical Data, Springer, New York, 1982.
[28] A. Blumer, A. Ehrenfreucht, D. Haussler, M. Warmuth, Learnability and the Vapnik–Chervonenkis dimension, Journal of the ACM 36 (1989) 929–965.
[29] A. Blum, On-line algorithms in machine learning, in: Proceedings of the Workshop on On-Line Algorithms, 1996, pp. 306–325.
[30] S. Sahni, Some related problems from network flows, game theory, and integer programming, in: Proceedings of the 13th Annual IEEE Symposium of Switching and Automata Theory, 1972, pp. 130–138.
[31] C.H. Papadimitriou, Computational Complexity, Addison-Wesley, Reading, MA, 1994.
[32] G. Ausiello, Complexity and Approximation: Combinatorial Problems and their Approximability, Springer-Verlag, Berlin, 1999.

[33] O. Ibarra, C.E. Kim, Fast approximation algorithms for the knapsack and sum of subset problems, Journal of the Association for Computing Machinery 22 (4) (1975) 463–468.
[34] W. Feller, An Introduction to Probability Theory and Its Applications, 2nd ed., vol. 1, John Wiley & Sons, 1950.
[35] F. Rosenblatt, The perceptron: A probabilistic model for information storage and organization in the brain, Psychological Review 65 (1959) 386–408.
[36] D.E. Rumelhart, Parallel Distributed Processing, Vol. 1, MIT Press, Cambridge, 1986.
[37] S. Kullback, Information Theory & Statistics, Wiley, 1959.
[38] J. Florens, M. Mouchart, J. Rolin, Elements of Bayesian Statistics, Marcel Dekker, Inc., New York, 1990.
[39] B. Efron, R. Tibshirani, An Introduction to the Bootstrap, Chapman and Hall, Freeman, New York, 1979.
[40] S. Cohen, N. Intrator, Forward and backward selection in regression hybrid network, Lecture Notes in Computer Science 2364 (2002) 98–107.