

Open Problems

1 When Is Small Beautiful?

Amiran Ambroladze¹ and John Shawe-Taylor²

The basic bound on the generalisation error of a PAC learner makes the assumption that a consistent hypothesis exists. This makes it appropriate to apply the method only in the case where we have a guarantee that a consistent hypothesis can be found, something that is rarely possible in real applications. The same problem arises if we decide not to use a hypothesis unless its error is below a prespecified number.

In this open problem we examine the implications of this fact for learning bounds.

Let $S = ((\mathbf{x}_1, y_1), \dots, (\mathbf{x}_l, y_l))$ be a training set, where the elements (\mathbf{x}_i, y_i) of S are generated independently and identically (i.i.d.) according to an unknown but fixed distribution over input/output pairings $(\mathbf{x}, y) \in X \times \{-1, 1\}$. Here y_i is called the label of the example \mathbf{x}_i . Let H be a hypothesis space, that is a collection of functions $h \in H$ mapping the input space X to the output space $\{-1, 1\}$. Denote: $\text{err}(h) := \text{Prob}\{(\mathbf{x}, y) : h(\mathbf{x}) \neq y\}$.

Fix $\epsilon > 0$. For the brevity we use the following terminology: A hypothesis h is *bad* if $\text{err}(h) > \epsilon$; otherwise h is *good*. We say that h is a *separator* for S if h is consistent with S : $h(\mathbf{x}_i) = y_i, i = 1, \dots, l$.

Denote $\delta(l) := \text{Prob}\{S : \exists \text{ a bad separator for } S\}$. It is easy to check that $\delta(l)$ is a monotonically decreasing function of l . This means that choosing big training sets is a winning strategy (at least theoretically) for getting better bounds for the classification error. In addition, if $d := \text{VCdim}(H) < \infty$ then we have (see [1]), that

$$\delta(l) < C(\epsilon, d) \cdot l^{2d} \cdot e^{-\epsilon l}, \quad (1)$$

where $C(\epsilon, d)$ depends only on ϵ and d . In particular, $\delta(l) \rightarrow 0$, as $l \rightarrow \infty$.

The smallness of $\delta(l)$ is partly guaranteed on the account of those training sets S for which there are no hypotheses $h \in H$ separating S . Some algorithms use an inference rule that selects a hypothesis $h \in H$ separating S , under the a-priori assumption that such a hypothesis exists for the given training set S . This makes relevant the following question about the conditional probability: How big is the probability of existence of a bad separator if we know that a separator (bad or good) exists? So, we are interested in the following quantity: $\delta(l)/\Delta(l) := \text{Prob}\{\exists \text{ a bad separator}\} / \text{Prob}\{\exists \text{ a separator}\}$.

¹ Lunds University, Lund

² Department of Computer Science, Royal Holloway, England

Some natural questions about this quotient are:

1) Is $\delta(l)/\Delta(l)$ still a monotonically decreasing function of l ?

2) Do we still have convergence to zero: $\delta(l)/\Delta(l) \rightarrow 0$ as $l \rightarrow \infty$?

In Example 1 below we have $\delta(l)/\Delta(l) \rightarrow 1$ and at the same time $\delta(l)/\Delta(l) < 1$. This implies that $\delta(l)/\Delta(l)$ does not decrease monotonically: there exists l_0 such that:

$$\delta(l_0)/\Delta(l_0) > \delta(l_0 - 1)/\Delta(l_0 - 1). \quad (2)$$

In this example $\text{VCdim}(H) = \infty$. In Example 2 we modify Example 1 so that $\text{VCdim}(H)$ becomes finite. We still have inequality (2), but it turns out that in this modified case we get

$$\delta(l)/\Delta(l) \rightarrow 0 \text{ as } l \rightarrow \infty. \quad (3)$$

Here a natural question arises: Does the condition $\text{VCdim}(H) < \infty$ always guarantee the limit relation (3)? A very trivial Example 3 below shows that this is not so. But Example 3 is ‘too’ trivial in the sense that there is no good hypothesis at all in that case; so trivially there $\delta(l) = \Delta(l)$ for all l . Now we formulate our

Conjecture: If $\text{VCdim}(H) < \infty$. Then a necessary and sufficient condition for (3) to be true is that H contains at least one good hypothesis.

Theorem 1. *If $\text{VCdim}(H) < \infty$ and if in addition H contains at least one hypothesis h , $\text{err}(h) < 1 - e^{-\epsilon}$, then we have (3).*

Note here that $1 - e^{-\epsilon} \approx \epsilon$ and by the definition the condition $\text{err}(h) < \epsilon$ means that h is good. This shows that the result of this theorem is very close to the statement in the conjecture. The proof of the theorem is essentially based on [1].

Theorem 1 shows that in most cases $\delta(l)/\Delta(l)$ decreases to zero, but we have no guarantee of monotonicity. This means that choosing big training sets for classification purposes may not always be a winning strategy.

Problem: Which conditions on the hypothesis space H guarantee that the conditional probability $\delta(l)/\Delta(l)$ is a monotonically decreasing function of l ?

Example 1. $X = (-1, 1)$, $\text{Prob}\{y = 1\} = \text{Prob}\{y = -1\} = 1/2$. The density functions for the positive and negative classes are $(x + 1)/2$ and $(1 - x)/2$, respectively. $\epsilon = 1/3$. For an arbitrary finite set $S = \{(\mathbf{x}_i, y_i)\}$ define $h_S(\mathbf{x}) = y_i$ if $\mathbf{x} = \mathbf{x}_i$, $\mathbf{x} \notin (-0.1, 0.1)$ and $h_S(\mathbf{x}) = 1$ otherwise. Let H consists of all such h_S (bad hypotheses) and plus the function $h_0 = \text{sgn}(x)$ (a good hypothesis).

Example 2. Here we define H in the same way as in Example 1, but take only those S which contain no more than l_0 elements. l_0 defined from (2).

Example 3. $X = \{1\}$, $\text{Prob}\{y = 1\} = \text{Prob}\{y = -1\} = 1/2$, $H = \{\text{sgn}(\mathbf{x})\}$.

References

- [1] Shave-Taylor J., Anthony M., Biggs N.L.: Bounding Sample Size with the Vapnik-Chervonenkis Dimension. *Discrete Applied Maths* **42** (1993) 65-73.

2 Learning a Function of r Relevant Variables

Avrim Blum^{3,4}

This problem has been around for a while but is one of my favorites. I will state it here in three forms, discuss a number of known results (some easy and some more intricate), and finally end with small financial incentives for various kinds of partial progress. This problem appears in various guises in [2, 3, 10]. To begin we need the following standard definition: a boolean function f over $\{0, 1\}^n$ has (at most) r relevant variables if there exist r indices i_1, \dots, i_r such that $f(x) = g(x_{i_1}, \dots, x_{i_r})$ for some boolean function g over $\{0, 1\}^r$. In other words, the value of f is determined by only a subset of r of its n input variables. For instance, the function $f(x) = x_1 \bar{x}_2 \vee x_2 \bar{x}_5 \vee x_5 \bar{x}_1$ has three relevant variables. The “class of boolean functions with r relevant variables” is the set of all such functions, over all possible g and sets $\{i_1, \dots, i_r\}$. The problems are:

- (a) Does there exist a polynomial time algorithm for learning the class of boolean functions that have $\lg(n)$ relevant variables, over the uniform distribution on $\{0, 1\}^n$?
- (b) Does there exist a polynomial time algorithm for learning the class of boolean functions that have $\lg \lg(n)$ relevant variables, over the uniform distribution on $\{0, 1\}^n$? Notice that since there are only $2^{2^{\lg \lg n}}$ possible g 's, we can assume the function g is known, and the only difficulty is determining which are the relevant variables.
- (c) Does there exist an algorithm for learning the class of boolean functions that have r relevant variables, over the uniform distribution on $\{0, 1\}^n$, in time “substantially” better than n^r ?

Motivation: These problems are all a special case of the question of whether DNF or Decision Trees can be learned in polynomial time. In particular, any function of r relevant variables can be written as a decision tree of depth r (branching on a different variable at each level) or as a DNF formula with at most 2^r terms (one for each positive entry in the truth-table for g). These both have polynomial size for $r = O(\log n)$, and so progress on the relevant-variable problem is necessary for any positive answers to those questions. Furthermore, the relevant-variable problem is arguably more basic than the DNF or Decision-Tree questions, since the target class is defined semantically rather than syntactically. Lastly, these problems suggest specific “challenge” distributions on target functions that could be used to test heuristics (see below).

Current Status: A recent and very nice result of Mossel, O’Donnell, and Servedio [10], building on ideas of Kalai and Mansour [9], achieves roughly $O(n^{0.7r})$ for problem (c). In the other direction, there exists a *specific* function g for which

³ Carnegie Mellon University

⁴ Supported in part by NSF grants CCR-0105488 and NSF-ITR CCR-0122581.

(if the set of relevant variables is chosen at random), all current techniques appear to break down at $O(n^{r/3})$ [2]. In addition to the question of whether [10] can be improved, a natural question is whether this specific case can be learned more efficiently, and whether one can construct “harder” functions g for which, say, beating $n^{r/2}$ appears hard. See below for rewards related to these statements.

Some Observations:

1. If membership queries are allowed, then learning is easy. (This makes a good homework question). Given a positive and negative example, one can “walk” them together to identify one relevant bit, put that at the top of a decision tree, and then recursively learn each subtree. Or, one can apply the more general algorithms of Bshouty [4] or Jackson [7].
2. If the target function is unbiased, then weak learning, strong learning, and exact identification are equivalent. (This also makes a good homework problem.) In particular, if A is a weak-learning algorithm, then one can identify a relevant variable by running A on data in which the first i bits of every example are replaced by new, truly random bits. If we do this for $i = 0, 1, \dots, n$, then at some point A must fail to perform better than random guessing, and this will occur at one of the relevant variables. If f is a *biased* function, then for this to work we need to change the definition of “weak learning” to mean an algorithm that performs noticeably better than the underlying bias of the target function.
3. We can assume without loss of generality that the relevant variables are chosen at random (since the algorithm can always randomly permute the indices if it so chooses).
4. Here is a specific function g proposed in [2] as a candidate hard case. Split the relevant variables into two sets A and B . On input x , compute the Parity function over A , and the Majority function over B , and then XOR the two results together.⁵ This class can be easily learned in time $O(n^{|A|})$ (by guessing the set A and reducing to majority) or in time $O(n^{|B|/2})$ (by guessing half of B and examining only the examples in which those bits are all 0, reducing to parity). The worst case is when $|A| = r/3, |B| = 2r/3$, yielding an algorithm that runs in time $O(n^{r/3})$, but no better algorithm is known.
 Notice that this specific function g gives a samplable distribution on target functions f (pick a random subset of r variables, split into A and B , and feed it into g). Thus one can test proposed heuristics.
5. Problems (a) and (b) are not solvable by SQ algorithms [8, 1]. This holds even for the specific g above. However, learning is easy for “most” functions g (e.g., if the truth table is picked at random). The difficult cases seem to be the functions g that are “similar to parity functions, but not exactly.”

⁵ For instance, if $A = \{1, 2, 3\}$ and $B = \{4, 5, 6\}$ then the classification of the example 011101001010 would be positive, since the first three bits have an even number of ones (making their parity 0), and the next three bits have more ones than zeros (so the majority function is 1), and the XOR of those two quantities is 1.

6. The theory of fixed-parameter tractability [5, 6] has been used to analyze the complexity of problems as a function of the size of the solution. For example, suppose we want to determine if an instance of the set-cover problem has a cover of size r . If there are n sets total, then it is easy to do this in time $O(n^r)$, but the results of [5] suggest that achieving running time of $f(r)poly(n)$ for any function f (e.g., $f(r) = 2^{2^r}$) may be hard. This has immediate consequences for the “proper learning” problem, if we allow examples to be arbitrary. For instance, it implies that determining if the data is consistent with a conjunction of size r is likely to be hard even if $r = O(\log n)$. However it is unclear whether this theory can be used to show (or suggest) hardness for the prediction problem.

Monetary Rewards:

\$100: Improve the results of [10] to $O(n^{0.666r})$.

\$200: Improve the results of [10] to $O(n^{0.499r})$.

\$100: Find an algorithm to learn the class described in Observation (4) in time $O(n^{r/4})$.

\$500: Find an algorithm to learn the class described in (4) in polynomial time, for $r = O(\log n)$.

\$50: Find a function g as in (4) but for which achieving $O(n^{r/2})$ appears hard.

\$1000: Give a positive solution to open problem (a) or (b).

\$50+: Give a convincing argument why nobody will ever be able to solve the above \$1000 problem. (Prize depends on how convincing.)

References

- [1] A. Blum, M. Furst, J. Jackson, M. Kearns, Y. Mansour, and S. Rudich. Weakly learning DNF and characterizing statistical query learning using fourier analysis. In *Proceedings of the 26th Annual ACM Symposium on Theory of Computing*, pages 253–262, May 1994.
- [2] A. Blum, M. Furst, M. Kearns, and D. Lipton. Cryptographic primitives based on hard learning problems. In D. Stinson, editor, *13th Annual International Cryptology Conference (CRYPTO)*. Springer, 1993. Lecture Notes in Computer Science No. 773.
- [3] Avrim Blum. Relevant examples and relevant features: Thoughts from computational learning theory. In *AAAI-94 Fall Symposium, Workshop on Relevance*, 1994.
- [4] N. H. Bshouty. Exact learning via the monotone theory. In *Proceedings of the IEEE Symposium on Foundation of Computer Science*, pages 302–311, Palo Alto, CA., 1993.
- [5] Rodney G. Downey and Michael R. Fellows. Fixed-parameter tractability and completeness i: Basic results. *SIAM Journal on Computing*, 24(4):873–921, 1995.
- [6] Rodney G. Downey and Michael R. Fellows. *Parameterized Complexity*. Monographs in Computer Science of Springer-Verlag, 1999.
- [7] J. Jackson. An efficient membership-query algorithm for learning DNF with respect to the uniform distribution. In *Proceedings of the IEEE Symposium on Foundation of Computer Science*, 1994.
- [8] M. Kearns. Efficient noise-tolerant learning from statistical queries. *Journal of the ACM*, 45(6):983–1006, 1998.
- [9] Adam Kalai and Yishay Mansour. Personal communication. 2001.
- [10] Elchanan Mossel, Ryan O’Donnell, and Rocco Servedio. Learning juntas. In *Proceedings of the 35th Annual ACM Symposium on Theory of Computing*, 2003.

3 Subspace Detection: A Robust Statistics Formulation

Sanjoy Dasgupta⁶

If data in \mathbb{R}^d actually lie in a linear subspace, then principal component analysis (PCA) will find this subspace. If the data are corrupted by benign (eg. independent Gaussian) noise, then approximation bounds can quite easily be shown for the solution returned by PCA. What if the noise is malicious?

ε -NOISY HYPERPLANE

Input: Points $x_1, \dots, x_n \in \mathbb{R}^d$

Output: A vector $w \in \mathbb{R}^d$ such that $w \cdot x_i = 0$ for at least $(1 - \varepsilon)$ fraction of the points

This is the typical sort of problem studied in the field of *robust statistics* ([2]). They typically consider data which has been corrupted by two types of noise: benign perturbation, as well as a small amount of malicious noise. The perturbation is generally easy to handle, and the main question is, how much malicious noise can be tolerated? The statistics literature offers two standard tricks.

1. Randomly sample d points, and hope they contain no outliers. Try this repeatedly, obtaining a candidate hyperplane each time; and return the best answer you get. The number of trials needed is something like $(1 - \varepsilon)^{-d}$, which is polynomial in d only for error rates $\varepsilon \leq O(\log d/d)$. Notice that this particular scheme assumes that the points are in general position (ie. any d good points specify the hyperplane).
2. Run PCA on the data to get a candidate hyperplane w ; throw away the point x with largest $|w \cdot x|$, and repeat. The behavior of this scheme has not been fully characterized, although Dunagan and Vempala ([1]) have analyzed something very similar.

There is a popular class of robust statistical estimators called M-estimators. It is known ([2]) that these can tolerate no more than a fraction $\varepsilon = \frac{1}{d+1}$ of outliers. Is it possible to do much better?

On the computational complexity side of things, the problem as stated above is NP-hard for any $\varepsilon = 1/\text{poly}(d)$; it is equivalent to being given a set of linear equalities and trying to satisfy as many of them as possible. However the reductions naively produce instances in which the points are not in general position.

References

- [1] J. Dunagan and S. Vempala. Optimal outlier removal in high-dimensional spaces. *Proceedings of the 32nd ACM Symposium on the Theory of Computing*, 2000.
- [2] P.J. Huber. *Robust statistics*. Wiley, 1984.

⁶ UC San Diego

4 How Fast Is k -Means?

*Sanjoy Dasgupta*⁷

The k -means algorithm is probably the most widely used clustering heuristic, and has the reputation of being fast. How fast is it exactly? Almost no non-trivial time bounds are known for it.

Let's say k -means is used to cluster n data points in \mathbb{R}^d . At any given time, it maintains a set of k "centers" μ_1, \dots, μ_k . These are often initialized by choosing randomly from the data set. The update rule is as follows:

1. Assign each data point to its closest center. This gives a Voronoi partitioning C_1, \dots, C_k of the data.
2. Reset each center μ_j to be the mean of (the data points in) its cluster C_j .

There is a particular cost function which strictly decreases on each step (if it doesn't, the algorithm halts):

$$\sum_{j=1}^k \sum_{x \in C_j} \|x - \mu_j\|^2,$$

where the inner summation is over the data points in cluster C_j .

The monotonic decrease of this cost function means that any particular configuration (a Voronoi k -partitioning) is never revisited, and therefore the number of iterations is at most the number of such partitionings (a simple upper bound is $n^{\binom{d+1}{k}}$). This is the only general bound known!

For the *one-dimensional* case I have checked the following.

1. For $k = 2$, there are instances which force $\Omega(n)$ iterations.
2. For $k < 5$, the number of iterations is at most $O(n)$.
3. If the ratio between the largest interpoint distance and the smallest interpoint distance is bounded by σ , then the number of iterations is $O(1/\sigma^2)$.

This appears to be the state of present knowledge.

Note that the quality of the final clustering is not the issue here: it is easy to show that it can be arbitrarily bad. The question is, how many iterations are needed until convergence?

⁷ UC San Diego

5 Universal Coding of Zipf Distributions

Yoav Freund,⁸ Alon Orlitsky,⁹ Prasad Santhanam,⁹ and Junan Zhang⁹

Background One of the best known results in information theory says that a data sequence x_1, x_2, \dots, x_n produced by independent random draws from a fixed distribution P over a discrete domain can be compressed into a binary sequence, or *code* whose expected length is at most $nH(P) + 1$ bits, where $H(P) = -\sum_i P_i \log P_i$ is the *entropy* of P . It is also known that this compression is near optimal as $nH(P)$ is the smallest achievable expected number of code bits.

In many data-compression applications the distribution P is not known except that it belongs to a given collection \mathcal{P} of distributions. In such situations the expected code length is longer and the difference between the expected code length and $nH(P)$ is the code's *redundancy*. Coding schemes whose redundancy increases slower than n are called *universal*.

For collections of distributions over finite alphabets, many results are known. For example, for the collection \mathcal{P}_k of iid distributions over an alphabet of size k it has been shown [1] that the optimal redundancy is $\frac{k-1}{2} \log n + o(\log n)$, hence iid distributions over finite alphabets can be universally compressed.

Considering these results from an online learning point of view, the set of distributions \mathcal{P} corresponds to a set of fixed experts (i.e. each expert makes a fixed prediction, independent of the past.) The universal coding scheme can be seen as an algorithm for combining expert advice, the code length is equal to the cumulative log loss (up to rounding error) and the redundancy is the difference between the loss of the combining algorithm and the loss of the best expert. See Freund [7] for an example of this type of analysis.

Large Alphabets When compressing natural-language text, it is reasonable to code the text a word at a time, thereby relying on the distribution of the words. In that case, the alphabet size k is the number of words in the language, which may be very large. Hence the $\frac{k-1}{2} \log n$ redundancy bound is not very meaningful.

One approach to reducing the redundancy in that case would be to restrict the collection of possible distributions over the words. The most popular way for characterizing such distributions is to sort the words in decreasing order of probability and characterize the probability of a word as a function of its place in the sorted list, also called its “rank”. Specifically, Zipf [2] observed that in many textual corpora, the distribution of words follows a power law whereby the probability of the i th ranked word is $P_i \approx \frac{C}{i^\alpha}$ where α is a constant close to 1 and C is a normalization factor.¹⁰ This observation underlies many applications of statistical natural language processing, see, e.g., [3].

⁸ Mitsubishi Electric Research Labs. Cambridge, MA

⁹ ECE Department, UC San Diego, La Jolla, CA 92093, e-mail: alon@ucsd.edu

¹⁰ For those that would like a more precise definition, a reasonable one might be to restrict P_i to be in the range $[\frac{C_1}{i^\alpha}, \frac{C_2}{i^\alpha}]$ where $C_2 > C_1 > 0$ are two constants.

Unknown Alphabets Another approach for compressing large, possibly infinite or even unknown, alphabets, is to decompose the description of a string into two parts: a description of the symbols, and of the order, or *pattern*, in which they appear. This is done for example, in the facsimile transmission of a document where the images corresponding to each character in the document are conveyed, then the order of the characters is transmitted.

This approach was taken in [4], and, in a slightly different context in [5]. Formally, the *index* $I(x)$ of a symbol x in a string \bar{x} is one more than the number of distinct symbols preceding x in \bar{x} , and the pattern of \bar{x} is the string $I(x_1), \dots, I(x_n)$ of indices. For example, in the string “abracadabra”, $I(a) = 1$, $I(b) = 2$, $I(r) = 3$, $I(c) = 4$, and $I(d) = 5$, hence the pattern of “abracadabra” is 12314151231.

It was shown in [6] that patterns of sequences generated by iid distributions can be universally compressed, regardless of the alphabet size. Specifically, that the redundancy of compressing the patterns of length- n sequences is $O(\sqrt{n})$.

Note however that this redundancy bound applies to patterns of strings generated by any iid distribution. As with standard compression, we would like to know whether there are lower-redundancy coding schemes for patterns of sequences generated by a distribution from a restricted class.

One such class is that of Zipf distributions with parameter α . It would be interesting to know whether one can achieve a redundancy bound of $O(n^\beta)$ where $\beta < 1/2$ and what is smallest achievable β as a function of α .

References

- [1] R.E. Krichevsky and V.K. Trofimov. The performance of universal coding. *IEEE Transactions on Information Theory*, 27:199–207, 1981.
- [2] G. Zipf. Selective studies and the principle of relative frequency in language. Technical report, Harvard university press, 1932.
- [3] W. Li. <http://linkage.rockefeller.edu/wli/zipf/>. *North Shore LIJ Research Institute*.
- [4] N. Jevtic, A. Orlitsky, and N.P. Santhanam. Universal compression of unknown alphabets. In *IEEE Symposium on Information Theory*, 2002.
- [5] J. Aberg, Y.M. Shtarkov, and B.J.M. Smeets. Multialphabet coding with separate alphabet description. In *Proceedings of compression and complexity of sequences*, 1997.
- [6] A. Orlitsky, N.P. Santhanam, and J. Zhang. Bounds on compression of unknown alphabets. In *IEEE Symposium on Information Theory*, 2003.
- [7] Y. Freund. Predicting a binary sequence almost as well as the optimal biased coin. *Information and Computation* 182 (2003) 73-94.

6 An Open Problem Regarding the Convergence of Universal A Priori Probability

Marcus Hutter^{11,12}

Abstract Is the textbook result that Solomonoff’s universal posterior converges to the true posterior for all Martin-Löf random sequences true?

Universal Induction Induction problems can be phrased as sequence prediction tasks. This is, for instance, obvious for time series prediction, but also includes classification tasks. Having observed data x_t at times $t < n$, the task is to predict the t -th symbol x_t from sequence $x = x_1 \dots x_{t-1}$. The key concept to attack general induction problems is Occam’s razor and to a less extent Epicurus’ principle of multiple explanations. The former/latter may be interpreted as to keep the simplest/all theories consistent with the observations $x_1 \dots x_{t-1}$ and to use these theories to predict x_t . Solomonoff [4, 5] formalized and combined both principles in his universal prior $M(x)$ which assigns high/low probability to simple/complex environments, hence implementing Occam and Epicurus. $M(x)$ is defined as the probability that a universal Turing machine U outputs a string starting with x , when provided with fair coin flips on the input.

Posterior Convergence Solomonoff’s [5] central result is that if the probability $\mu(x_t|x_1 \dots x_{t-1})$ of observing x_t at time t , given past observations $x_1 \dots x_{t-1}$ is a computable function, then the universal posterior $M_t := M(x_t|x_1 \dots x_{t-1})$ converges (rapidly!) with μ -probability 1 (w.p.1) for $t \rightarrow \infty$ to the true posterior $\mu_t := \mu(x_t|x_1 \dots x_{t-1})$, hence M represents a universal predictor in case of unknown μ . Convergence of M_t to μ_t w.p.1 tells us that M_t is close to μ_t for sufficiently large t for “almost all” sequences $x_{1:\infty}$ (we abbreviate $x_{1:n} := x_1 \dots x_n$). It says nothing about whether convergence is true for any *particular* sequence (of measure 0).

Martin-Löf Randomness is the standard notion to capture convergence for individual sequences and is closely related to Solomonoff’s universal prior. Levin gave a characterization equivalent to Martin-Löf’s (M.L.) original definition [2]:

A sequence $x_{1:\infty}$ is μ -random (in the sense of M.L.) iff there is a constant c such that $M(x_{1:n}) \leq c \cdot \mu(x_{1:n})$ for all n .

One can show that a μ -random sequence $x_{1:\infty}$ passes *all* thinkable effective randomness tests, e.g. the law of large numbers, the law of the iterated logarithm, etc. In particular, the set of all μ -random sequences has μ -measure 1.

Open Problem An interesting open question is whether M_t converges to μ_t (in difference or ratio) individually for all Martin-Löf random sequences. Clearly, Solomonoff’s result shows that convergence may at most fail for a set of sequences with μ -measure zero. A convergence result for μ -random sequences is particularly interesting and natural in this context, since μ -randomness can be defined in terms of M itself (see above).

¹¹ IDSIA, Galleria 2, CH-6928 Manno-Lugano, Switzerland
marcus@idsia.ch <http://www.idsia.ch/~marcus>

¹² A prize of 128 Euro for a solution of this problem is offered.

Proof Attempts Attempts to convert the convergence results w.p.1 to effective μ -randomness tests fail, since M_t is not lower semi-computable. In [3, Th.5.2.2] and [6, Th.10] the following Theorem is stated:

“Let μ be a positive recursive measure. If the length of y is fixed and the length of x grows to infinity, then $M(y|x)/\mu(y|x) \rightarrow 1$ with μ -probability one. The infinite sequences ω with prefixes x satisfying the displayed asymptotics are precisely [\Leftarrow] and [\Leftarrow] the μ -random sequences.”

While convergence w.p.1 is correct if appropriately interpreted,¹³ the proof that convergence holds for μ -random sequences is incomplete:

“ $M(x_{1:n}) \leq c \cdot \mu(x_{1:n}) \forall n \Rightarrow \lim_{n \rightarrow \infty} M(x_{1:n})/\mu(x_{1:n})$ exists” has been used, but not proven, and may indeed be wrong. Vovk [7] shows that for two finitely computable semi-measures μ and ρ , and $x_{1:\infty}$ being μ - and ρ -random that $\rho_t/\mu_t \rightarrow 1$. If M were recursive, then this would imply $M_t/\mu_t \rightarrow 1$ for every μ -random sequence $x_{1:\infty}$, since every sequence is M -random. Since M is *not* recursive Vovk’s theorem cannot be applied and it is not obvious how to generalize it. So the question of individual convergence remains open.

Conclusions Contrary to what was believed before, the question of posterior convergence $M_t/\mu_t \rightarrow 1$ (also $M_t \rightarrow \mu_t$) for all μ -random sequences is still open. In [1] we introduce a new flexible notion of randomness which contains Martin-Löf randomness as a special case. This notion is used to show that standard proof attempts of $M_t/\mu_t \xrightarrow{M.L.} 1$ based on so called dominance only must fail, indicating that this problem may be a hard one.

References

- [1] M. Hutter. On the existence and convergence of computable universal priors. Technical Report IDSIA-05-03, 2003. <http://arxiv.org/abs/cs.LG/0305052>.
- [2] L. A. Levin. On the notion of a random sequence. *Soviet Math. Dokl.*, 14(5):1413–1416, 1973.
- [3] M. Li and P. M. B. Vitányi. *An introduction to Kolmogorov complexity and its applications*. Springer, 2nd edition, 1997.
- [4] R. J. Solomonoff. A formal theory of inductive inference: Part 1 and 2. *Inform. Control*, 7:1–22, 224–254, 1964.
- [5] R. J. Solomonoff. Complexity-based induction systems: comparisons and convergence theorems. *IEEE Trans. Inform. Theory*, IT-24:422–432, 1978.
- [6] P. M. B. Vitányi and M. Li. Minimum description length induction, Bayesianism, and Kolmogorov complexity. *IEEE Transactions on Information Theory*, 46(2):446–464, 2000.
- [7] V. G. Vovk. On a randomness criterion. *Soviet Mathematics Doklady*, 35(3):656–660, 1987.

¹³ The formulation of the Theorem is quite misleading in general: First, for off-sequence y convergence w.p.1 does not hold (xy must be demanded to be a prefix of $x_{1:\infty}$). Second, the proof of [\Leftarrow] has loopholes (see main text). Last, [\Rightarrow] is given without proof and is probably wrong. Also the assertion in [3, Th.5.2.1] that S_t converges to zero faster than $1/t$ cannot be made, since S_t may not decrease monotonically.

7 Entropy Bounds for Restricted Convex Hulls

Vladimir Koltchinski¹⁴

Abstract An unsolved problem of bounding the entropy of a "restricted" convex hull of a set in a Hilbert space is discussed. The problem is related to bounding the generalization error of convex combinations of base classifiers.

Let H be a separable Hilbert space and $K \subset H$. Without loss of generality, one can assume that H is the space ℓ_2 of all square summable sequences equipped with the canonical inner product. Let $\text{conv}(K)$ denote the convex hull of K :

$$\text{conv}(K) := \left\{ \sum_j \lambda_j x_j : x_j \in K, \lambda_j \geq 0, \sum_j \lambda_j = 1 \right\}.$$

For a set $T \subset H$, $N(T; \varepsilon)$ denotes the minimal number of balls of radius $\varepsilon > 0$ covering T and let $H(T; \varepsilon) := \log N(T; \varepsilon)$ be the ε -entropy of T . We assume in what follows that for some $V > 0$ $N(K; \varepsilon) = O(\varepsilon^{-V})$ as $\varepsilon \rightarrow 0$.

Then, it is well known (see, e.g., [1], Theorem 2.6.9) that

$$H(\text{conv}(K); \varepsilon) = O\left(\varepsilon^{-2V/(2+V)}\right) \text{ as } \varepsilon \rightarrow 0.$$

Convex hulls are of importance in learning theory since boosting type algorithms output combined classifiers from the convex hull of a given base class of functions. The analysis of such algorithms and, especially, the development of subtle probabilistic bounds on their generalization error often require the knowledge of not only the entropy of the whole convex hull, but also the entropy of its subsets defined in terms of various restrictions on the coefficients of convex combinations as well as on the vectors involved in them. We describe below an important version of this problem.

Given $x \in \text{conv}(K)$, define

$$\mathcal{N}(x, \varepsilon) := \inf \left\{ N(K'; \varepsilon), K' \subset K, x \in \text{conv}(K') \right\}.$$

Clearly, if $x = \sum_{j=1}^m \lambda_j x_j$, then

$$\mathcal{N}(x, \varepsilon) \leq N(\{x_1, \dots, x_m\}; \varepsilon).$$

So, if x is a convex combination of a base classifiers x_j , $\mathcal{N}(x; \varepsilon)$ can be bounded by the ε -covering number of the set $\{x_j\}$ of base classifiers.

If now N is a nonincreasing nonnegative function on $(0, +\infty)$, let us define

$$\text{conv}_N(K) := \left\{ x \in \text{conv}(K) : \forall \varepsilon > 0 : \mathcal{N}(x, \varepsilon) \leq N(\varepsilon) \right\}.$$

¹⁴ Department of Mathematics and Statistics, University of New Mexico, Albuquerque, NM, 87131-1141, USA

Problem: Find a sharp upper bound on the ε -entropy of the set $\text{conv}_N(K)$.
It is easy to prove that

$$H\left(\text{conv}_N(K); \varepsilon\right) = O(N(\varepsilon) \log(1/\varepsilon)) \text{ as } \varepsilon \rightarrow 0,$$

which is sharp if $N(\varepsilon) \equiv d$, d is a constant. However, if $N(\varepsilon) = C\varepsilon^{-V} \geq N(K; \varepsilon)$, then $\text{conv}_N(K) = \text{conv}(K)$, and the above bound becomes much worse than $O\left(\varepsilon^{-2V/(2+V)}\right)$, which also holds in this case. So, the problem is to find a subtle interpolation between these two cases.

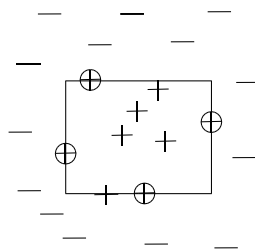
References

- [1] van der Vaart, A.W. and Wellner, J. Weak Convergence and Empirical Processes.
Springer, New York (1996)

8 Compressing to VC Dimension Many Points

Manfred K. Warmuth¹⁵

Any set of labeled examples consistent with some hidden orthogonal rectangle can be “compressed” to at most four examples: An upmost, a leftmost, a rightmost and a bottommost positive example. These four examples represent an orthogonal rectangle (the smallest such rectangle that contains them) that is consistent with all examples.



Note that the VC dimension of orthogonal rectangles is four and this is exactly the number of examples needed to represent the consistent orthogonal rectangle.

A compression scheme of size k for a concept class C picks from any set of examples consistent with some concept in C a subset of up to k examples and this subset represents (via a mapping that is specific to the class C) a hypothesis consistent with the whole original set of examples.

Conjecture: Any concept class of VC dimension d has a compression scheme of size d .

What evidence do we have that this conjecture might be true?

Call a concept class of VC dimension d maximum if for every subset of m instances, the concept class induces exactly $\sum_{i=1}^d \binom{m}{i}$ concepts on the subset. That is, for every subset of m instances, we have exactly the maximum number of induced concepts so that Sauer’s lemma is tight.

In [1] it was shown that every maximum concept class of VC dimension d has a compression scheme that compresses example sets of size larger than d to a subset of exactly d examples.

Compression schemes were introduced in [2]. In that note a PAC bound is given (See also appendix of [1]). The size of the compression scheme takes the role of the VC dimension in the PAC bound and the constants are small. The proof is very short and interesting in its own right.

Various variants of the above definition of compression scheme are discussed in [2] and [1], Section 9.

Monetary rewards: \$600 for resolving the conjecture by either proving that for any concept class of VC dimension d there always is a compression scheme of size $O(d)$, or by providing a counter example that shows that such compressions schemes are not always possible.

¹⁵ UC Santa Cruz

References

- [1] S. Floyd and M. K. Warmuth. Sample compression, learnability, and the Vapnik-Chervonenkis dimension. *Machine Learning*, 21(3):269–304, 1995.
- [2] N. Littlestone and M. K. Warmuth. Relating data compression and learnability. Unpublished manuscript, obtainable at <http://www.cse.ucsc.edu/~manfred>, June 10 1986.