Machine Learning 6: Computational Complexity of Learning

Master 2 Computer Science

Aurélien Garivier

2018-2019

Table of contents

Computational Complexity of Learning

Computational complexity of a learning algorithm 1/2

Definition

An algorithm $\mathcal{A}$ solves the learning task with domain set $\mathcal{X} \times \mathcal{Y}$, hypothesis class $\mathcal{H}$ and 0-1 loss in time $O(f)$ if there exists some constant $c > 0$ such that for every probability distribution D over $\mathcal{X} \times \mathcal{Y}$, and every $\epsilon, \delta > 0$, when $\mathcal{A}$ receives as input iid samples of D:

- $\mathcal{A}$ terminates after performing at most $cf(\epsilon, \delta)$ operations,
- the output of $\mathcal{A}$, denoted by $h_{\mathcal{A}}$, can perform a prediction on a new datapoint by performing at most $cf(\epsilon, \delta)$ operations,
- $h_{\mathcal{A}}$ is $(\epsilon, \delta)-$PAC: with probability at most $1 - \delta$, $P_D(h_{\mathcal{A}}(X) \neq Y) \leq \min_{h \in \mathcal{H}} \mathbb{P}_D(h(X) \neq Y) + \epsilon$.

NB: the second point is to ensure that the learning process is not "hidden" in the prediction function.

Computational complexity of a learning algorithm 2/2

Definition

A sequence $\mathcal{X}_n \times \mathcal{Y}_n$, $\mathcal{H}_n$ of learning problems is solved by algorithm $\mathcal{A}$ in time $O(g)$, where $g : \mathbb{N} \times (0,1)^2 \to \mathbb{N}$, if for all n $\mathcal{A}$ solves the task $(\mathcal{X}_n \times \mathcal{Y}_n, \mathcal{H}_n, \ell_n)$ in time $O(f_n)$, where $f_n : (0,1)^2 \to \mathbb{N}$ is defined by $f_n(\epsilon, \delta) = g(n, \epsilon, \delta)$.

NB: in this definition the constant c of the $O(f)$ may depend on n.

$\mathcal{A}$ is *efficient* if one can choose g polynomial (wrt all variables).

Example: a finite hypothese class has polynomial sample complexity, but the ERM can be long to find if $\left|\mathcal{H}_n\right|$ is not polynomial in n.

Learning Boolean functions

Boolean conjunctions

For a positive integer n, $1 \leq k, r \leq n$ and $1 \leq i_1, \dots, i_k, j_1, \dots, j_r \leq n$, the boolean conjunction

$$x_{i_1} \wedge \cdots \wedge x_{i_k} \wedge \neg x_{j_1} \wedge \cdots \wedge \neg x_{j_r}$$

defines the function $h : \mathcal{X} = \{0,1\}^n \to \mathcal{Y} = \{0,1\}$ by

$$h(x) = \begin{cases} 1 & \text{if } x_{i_1} = \cdots = x_{i_k} = 1 \text{ and } x_{j_1} = \cdots = x_{j_r} = 0 \,, \\ 0 & \text{otherwise.} \end{cases}$$

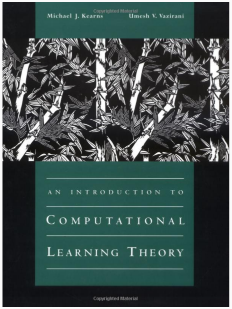
The class of all boolean functions over $\{0,1\}^n$ is denoted by $\mathcal{H}_C^n$, and has size at most $3^n + 1$. Hence, its sample complexity is at most $\dfrac{n \log(3/\delta)}{\epsilon}$.

Learning boolean conjunctions

Theorem

- In the realizable case, it is possible to compute an ERM in time $O(mn)$.
- Unless P=NP, there is no algorithm running in time polynomial in n and m that is guaranteed to find an ERM hypothesis in the agnostic case.

Reference: *An Introduction to Computational Learning Theory*,
by Michael J. Kearns and Umesh Vazirani,
MIT Press (1994).



Learning 3-term DNF

The class $\mathcal{H}^n_{3DNF}$ of 3-term Disjunctive Normal Form formulas is made of the boolean functions of the form

$$h(x) = A_1(x) \vee A_2(x) \vee A_3(x) \ ,$$

where the $A_j(x)$ are boolean conjunctions. It has size at most 3^{3n} and is thus learnable with sample complexity at most $3n \log(3/\delta)/\epsilon$. But from a computational perspective, even the realizable case is hard.

Theorem

Unless RP=NP, no algorithm *properly* learns a sequence of 3-term DNF problems in polynomial time.

Idea: if you can properly learn 3-term DNF, you have random algorithm able to compute an ERM whp by taking $\epsilon = 1/(2m)$ and $D =$ uniform distribution on the sample; but computing an ERM is NP-hard (see next slide).

Theorem

There exists a *representation independent* learning algorithm for 3-term DNF problems in time $O(n^3 m)$.

Proof: computing an ERM is NP-hard

Idea: reduction of the graph 3-coloring problem. A graph $G = (V, E)$ is 3-colorable if there exists a mapping $f : V \to \{1, 2, 3\}$ such that $(u, v) \in E \implies f(u) \neq f(v)$.

Assume that an algorithm computes an ERM for $\mathcal{H}$ in polynomial time in n and m. For any graph $G = (V, E)$, where $V = \{1, \ldots, n\}$, let $m = |V| + |E|$ and $S \in (\mathbb{R}^n \times \{0, 1\})^m$ be the sample containing:

- for every $i \in \{1, \ldots, n\}$, the pair $(e_{-i}, 1)$, where $e_{-i} = (1, \ldots, 1) - e_i$;
- for every edge $(i, j) \in E$, the pair $\big(e_{-ij}, 0\big)$, where $e_{-ij} = e_{-j} - e_i$.

Then:

- if there exists $h \in \mathcal{H}$ that has zero error on S, then G is 3-colorable: take $f(i) = \min\{c : A_c(e_{-i}) = 1\}$. If $f(i) = f(j) = c$, $A_c(e_{-i}) = A_c(e_{-j}) = 1$; but $(e_{-i})_i = 0$ whereas $(e_{-j})_i = 1$, hence A_c does not involve x_i: as e_{-ij} differs from e_j just at x_i, $A_c(e_{-ij}) = 1$ and by construction $(i, j) \notin E$.
- if G is 3-colorable, then there exists $h \in \mathcal{H}$ with zero error on S: for $c \in \{1, 2, 3\}$ take $A_c(x) = \bigwedge_{i : f(i) \neq c} x_i$; then $h(e_{-i}) = A_{f(i)}(e_{-i}) = 1$ and if $(i, j) \in E$, $f(i) \neq f(j)$ implies $A_c(e_{-ij}) = 0$ for all c.

Proof: 3-term DNF are representation-independent learnable

For every $x \in \{0,1\}^n$ let $u^x \in \{0,1\}^{2n}$ by $u_i^x = x_i$ if $1 \leq i \leq n$ and $u_i^x = \neg x_i$ if $n+1 \leq i \leq 2n$. We write, for $c \in \{1,2,3\}$, $A_c = \bigwedge_{u \in A_c} u$. Since $\vee$ distributes over $\wedge$,

$$h(x) = \bigwedge_{u_1 \in A_1, u_2 \in A_2, u_3 \in A_3} u_1 \vee u_2 \vee u_3 \ . \tag{1}$$

Let $\psi : \{0,1\}^n \to \{0,1\}^{(2n)^3}$ be such that $\left[\psi(x)\right]_{i_1,i_2,i_3} = u_{i_1}^x \vee u_{i_2}^x \vee u_{i_3}^x$. By Equation (1), there exists a conjunction $H : \{0,1\}^{(2n)^3} \to \{0,1\}$ such that for every $x \in \{0,1\}^n$, $h(x) = H(\psi(x))$.

Hence, since we saw earlier that conjunctions of $(2n)^3$ variables are efficiently learnable with sample complexity at most $n^3 \log(1/\delta)/\epsilon$, there exists an algorithm computing a function $\hat{H} : \{0,1\}^{(2n^3)} \to \{0,1\}$ compatible with all the examples $\Big\{\big(\psi(x), y\big) : (x,y) \in S\Big\}$ in $O(mn^3)$ operations. It permits to define $\hat{h} : \{0,1\}^n \to \{0,1\}$ by $\hat{h}(x) = \hat{H}\big(\psi(x)\big)$, which agrees with all samples: it is an ERM. This does not contradict the NP-hardness result above: $\hat{h}$ is generically not a 3-term DNF.

Learning axis-aligned rectangles

Theorem

Let $\mathcal{H}^n_{\mathrm{rec}} = \left\{ h_{(a_1,b_1,\ldots,a_n,b_n)} : a_1 \le b_1, \ldots, a_n \le b_n \right\}$ where

$$h_{(a_1,b_1,\ldots,a_n,b_n)}(x_1,\ldots,x_n) = \begin{cases} 1 & \text{if } a_1 \le x_1 \le b_1, \ldots, a_n \le x_n \le b_n\,; \\ 0 & \text{otherwise}\,. \end{cases}$$

- In the realizable case, an ERM can be computed in $O(nm)$ operations: pick $a_i = \min\left\{x_i : (x,1) \in S\right\}$ and $b_i = \max\left\{x_i : (x,1) \in S\right\}$.
- In the agnostic case, solving the ERM is NP-hard: unless P=NP, there is on algorithm whose running time is polynomial in m and n that is guaranteed to find an ERM.
- However, for a fixed dimension n, the ERM can be computed in polynomial time in m (try all subsets of the sample of size $2n$).

Research Article 3

On the difficulty of approximately maximizing agreements

by Shai Ben-David, Nadav Eiron and Philip M. Long

Journal of Computer and System Sciences Vol. 66.3, May 2003, pp. 496–514

`https://dl.acm.org/citation.cfm?id=859224`

On the Difficulty of Approximately Maximizing Agreements[*]

Shai Ben-David
Department of Computer Science
Technion
Haifa 32000, Israel
shai@cs.technion.ac.il

Nadav Eiron
IBM Almaden Research Center
650 Harry Road
San Jose, CA 95120 USA
nadav@us.ibm.com

Philip M. Long
Genome Institute of Singapore
1 Science Park Road
The Capricorn, #05-01
Singapore 117604, Republic of Singapore
gislongp@nus.edu.sg

October 8, 2002

Abstract

We address the computational complexity of learning in the agnostic framework. For a variety of common concept classes we prove that, unless P=NP, there is no polynomial time approximation scheme for finding a member in the class that approximately maximizes the agreement with a given training sample. In particular our results apply to the classes of monomials, axis-aligned hyper-rectangles, closed balls and monotone monomials. For each of these classes we prove the NP-hardness of approximating maximal agreement to within some fixed constant (independent of the sample size and of the dimensionality of the sample space). For the class of half-spaces, we prove that, for any $\epsilon > 0$, it is NP-hard to approximately maximize agreements to within a factor of $(418/415 - \epsilon)$, improving on the best previously known constant for this problem, and using a simpler proof.

An interesting feature of our proofs is that, for each of the classes we discuss, we find patterns of training examples that, while being hard for approximating agreement within that concept class, allow efficient agreement maximization within other concept classes. These results bring up a new aspect of the model selection problem – they imply that the choice of hypothesis class for agnostic learning from among those considered in this paper can drastically effect the computational complexity of the learning process.

Keywords: Machine learning, computational learning theory, neural networks, inapproximability, hardness, half-spaces, axis-aligned hyper-rectangles, balls, monomials.

[*]An early version of this paper appeared in the Proceedings of the Thirteenth Annual Conference on Computational Learning Theory, 2000.

1