

# Alternative Measures of Computational Complexity with Applications to Agnostic Learning

Shai Ben-David

School of Computer Science  
University of Waterloo,  
Waterloo, Ontario, N2L 3G1, Canada  
shai@cs.uwaterloo.ca

**Abstract.** We address a fundamental problem of complexity theory - the inadequacy of worst-case complexity for the task of evaluating the computational resources required for real life problems. While being the best known measure and enjoying the support of a rich and elegant theory, worst-case complexity seems gives rise to over-pessimistic complexity values. Many standard task, that are being carried out routinely in machine learning applications, are NP-hard, that is, infeasible from the worst-case-complexity perspective. In this work we offer an alternative measure of complexity for approximations-optimization tasks. Our approach is to define a hierarchy on the set of inputs to a learning task, so that natural ('real data') inputs occupy only bounded levels of this hierarchy and that there are algorithms that handle in polynomial time each such bounded level.

## 1 Introduction

Computational complexity theory aims to provide tools for the quantification and analysis of the computational resources needed for algorithms to perform computational tasks. Worst-case complexity is by far the best known, most general, most researched and best understood approach in computational complexity theory. However, it is becoming apparent that this measure is unrealistically pessimistic. As a conspicuous example one may consider the satisfiability problem for propositional logic, SAT. Being NP complete, SAT is infeasible from the point of view of worst-case complexity. Just the same, SAT solvers, programs that efficiently solve large instances of SAT, are becoming more and more popular and useful as a practical tool for solving large scale real life problems. Similar phenomena occurs in many other areas. In particular, in the machine learning domain, empirical risk minimization (i.e., the minimization of training error) have been shown to be NP-hard to approximate for practically all common learning classes, yet many machine learning tools solve such problems on a regular basis.

The reason for this phenomena is pretty obvious. The worst-case complexity of a problem, as the name suggests, is determined by the hardest instances.

If a problem is, say, exponential time hard, it means that for every algorithm that answers correctly on *all* inputs, *there exist* an infinite sequence of inputs on which it will run for exponential number of steps. However, it does not rule out the possibility that these hard inputs are very sparsely scattered and that in practice one never encounters any instance that requires a long run of the algorithm.

This raises the following question - is there a way to measure the complexity of problems in a manner that is more relevant to our actual experience with these problems? Can one define complexity measures that reflects the hardness of tasks on real data?

In this work, we wish to propose an alternative approach to measuring the computational complexity of optimization tasks. The new approach defines (implicitly) a measure of "naturalness" of an input. The intention is that one hand it is reasonable to assume that reality-generated instances satisfy this "naturalness" property and that, on the other hand, there exist algorithms that solve all "natural" instances in polynomial time (for problems that may be NP hard).

We focus our attention on optimization problems. In these problems, there is an objective function that associates a real-valued cost with every input-solution pair. Our approach can be viewed as measuring the robustness of an input in terms of the objective function. Namely, inputs are considered "natural" if the value of the objective function for their optimal solutions does not change dramatically when these optimal solution are mildly perturbed.

The over-pessimism of worst case complexity bounds has been addressed before. Average case complexity [1], [2], as well as parameterized complexity [3], offer alternative measure of complexity that provide some partial answers to this question. The work of Spielman and Teng on Smoothed Analysis [4] is probably the most prominent approach explicitly aimed at answering the concerns described above. Roughly speaking, they propose to perturb inputs and replace the complexity of solving each particular input, by the average complexity over its perturbed images. We take quite a different approach, focussing on finding features that distinguish real-life inputs from arbitrary theoretically-constructed ones.

## 2 Notation and Definitions

### 2.1 Some basic combinatorial optimization terminology

A combinatorial optimization problem is define by a triple  $\mathcal{P} = (\mathcal{I}, \mathcal{T}, \Pi)$  where  $\mathcal{I}$  is the set of possible inputs,  $\mathcal{T}$  is the set of possible solutions and  $\Pi : \mathcal{I} \times \mathcal{T} \mapsto \mathbb{R}^+$  is a gain (or loss) function. An optimization algorithm for  $\mathcal{P}$ ,  $A$ , maps inputs to solutions. For concreteness, let us focus on maximization problems, where the goal of an algorithm is to find, for an input  $I$ , a solution  $T$  that maximizes the gain  $\Pi(I, T)$ . We denote by  $\text{Opt}_{\mathcal{P}}$  the function that maps each input  $I \in \mathcal{I}$  to  $\sup\{\Pi(I, T) : T \in \mathcal{T}\}$  (we omit the subscript  $\mathcal{P}$  when it is clear from the context). We assume, throughout this paper, that this supremum is achievable,

and denote by  $T_{opt}(I)$  the solution that achieves this maximum value (so, for every  $I$ ,  $\Pi(I, T_{opt}(I)) = Opt(I)$ ).

**Definition 1.** 1. The relative loss of an algorithm  $A$  on an input  $I$  is defined as

$$\frac{Opt(I) - \Pi(I, A(I))}{Opt(I)}$$

2. For  $\epsilon \leq 1$ , we say that  $A$  is a  $\epsilon$ -approximation algorithm for  $\mathcal{P}$ , if its relative loss, is at most  $\epsilon$ , Namely

$$\Pi(I, A(I)) \geq \sup_{T \in \mathcal{T}} \Pi(I, T)(1 - \epsilon)$$

for every  $I \in \mathcal{I}$ .

It turns out that for many natural approximation problems the task of finding good approximations is computationally infeasible. Namely, assuming  $P \neq NP$ , for some  $\epsilon > 0$ , no  $\epsilon$ -approximation algorithm runs in polynomial time. We say that such problems are NP-hard to approximate.

## 2.2 Combinatorial approximation problems in machine learning

We shall consider a family of optimization problems that arise when one wishes to carry out Empirical Risk Minimization in the context of agnostic learning. Namely,

**The Maximal Agreement Problem for a concept class:** Given a concept class  $\mathcal{H}$  over some domain set  $\mathcal{X}$  (so,  $\mathcal{H} \subseteq \{B : B \subseteq X\}$ ), the maximum agreement problem for  $\mathcal{H}$  is defined by having as inputs the set of all finite labeled samples  $\mathcal{I} = \{(x_1, \eta_1), \dots, (x_m, \eta_m)\} : m \in \mathbb{N}, x_i \in X \text{ and } \eta_i \in \{-1, 1\}$ , the set of solutions  $\mathcal{T} = \mathcal{H}$ , and the gain is defined by

$$\Pi(S, h) = \frac{|\{(x, \eta) \in S : h(x) = \eta\}|}{|S|}$$

In particular, we shall consider

**Best Separating Hyper-plane (BSH)** For some  $n \geq 1$ , inputs are of the form  $S = \{(x_1, \eta_1), \dots, (x_m, \eta_m)\}$ , where  $(x_i, \eta_i) \in \mathbb{R}^n \times \{+1, -1\}$  for all  $i$ . For a hyper-plane  $h(w, t)$ , where  $w \in \mathbb{R}^n$  and  $t \in \mathbb{R}$ ,  $h(x) = \text{sign}(\langle w, x \rangle - t)$  where  $\langle w, x \rangle$  denotes the dot product of the vectors  $w$  and  $x$ .

The goal of a *Best Separating Hyper-plane algorithm* is to find a pair  $(w, t)$  so that  $\Pi(S, h(w, t))$  is as large as possible.

**Best Separating Homogeneous Hyper-plane (BSHH)** Is the same problem as BSH, except that we restrict the search to homogeneous hyper-planes.

**Densest Open Ball (DOB)** For some  $n \geq 1$ , inputs are lists of points from  $\mathbb{R}^n$ ,  $P = (x_1, \dots, x_m)$ . The problem is to find the *Densest Open Ball* of radius 1 for  $P$ . Formally,  $\mathcal{T} = \mathbb{R}^n$ , and  $\Pi(P, z) = |\{i : x_i \in B(z, 1)\}|$ , where the  $x_i$ 's are the points listed in  $P$  and  $B(z, 1)$  is the  $n$  dimensional ball of radius 1 centered at  $z$ .

**Theorem 1** ([5]). *The Best Separating Hyperplane problem, as well as its homogenous version BSHH, are NP-hard to approximate.*

**Theorem 2** ([6]). *The Densest Open Ball problem is, NP-hard to approximate.*

### 3 A Different measure of approximation

#### 3.1 definition and intuition

We propose to add a measure of proximity between solutions. That is, a function  $d: \mathcal{T} \times \mathcal{T} \mapsto \mathbb{R}^+$  so that

$$\forall T, T' d(T, T') = d(T'T) \text{ and } d(T, T') = 0 \text{ iff } T = T'$$

Examples of such natural proximity measures are the distance between ball centers for the DOB problem, namely,  $d(B(z, 1), B(z', 1)) = |z - z'|$ , any common distance between hyper-planes for the BSH problem (say, the  $L_2$  distance over the  $n + 1$  dimensional spaces of solutions  $(w, t)$ , or  $d(w, w') = 1 - \langle w, w' \rangle$  for homogenous hyper-planes of unit weight).

**Definition 2.** 1. For  $I \in \mathcal{I}, T \in \mathcal{T}$ ,

$$\lambda_{(I, T)} \stackrel{\text{def}}{=} \sup_{T' \in \mathcal{T}} \frac{\Pi(I, T) - \Pi(I, T')}{\Pi(I, T)d(T, T')}$$

*I.e.,  $\lambda_{(I, T)}$  measures the rate by which the gain function changes (normalized by its value at  $T$ ) as  $d(T, T')$  grows (in other words, the slope of the gain function for  $I$  around  $T$ ).*

2. For  $\rho > 0$  let

$$\lambda_{(I, T)}^\rho \stackrel{\text{def}}{=} \sup_{T' \in \mathcal{T}, d(T, T') \leq \rho} \frac{\Pi(I, T) - \Pi(I, T')}{\Pi(I, T)d(T, T')}$$

*The only difference here is that we only care about the slope of the gain function in the  $\rho$ -neighborhood of  $T$ .*

3. An algorithm  $A$  is a  $\rho$ -approximation for an optimization problem  $\mathcal{P}$  w.r.t a proximity measure  $d$  if, for every input  $I$ ,

$$\Pi(I, A(I)) \geq \sup_{T \in \mathcal{T}} \Pi(I, T)(1 - \rho\lambda_{(I, T)}^\rho)$$

Note that that  $\lambda^\rho$  is a monotone function of  $\rho$ . It therefore follows that so is  $\rho\lambda^\rho$ . Consequently, the approximation improves as  $\rho$  shrinks.

Recall that the usual  $\epsilon$ -approximation definition requires  $\Pi(I, A(I)) \geq \sup_{T \in \mathcal{T}} \Pi(I, T)(1 - \epsilon)$ . So, rather than approximating the optimal solution to within some fixed  $\epsilon$  fraction of the gain of the optimal solution, we require an approximation margin that depends on a property of the input  $I$  - the slope of the gain function for  $I$ . It allows a loose approximation for inputs all of whose close-to-optimal solutions are very frail - small perturbations of these solutions result in a sharp deterioration of their gain.

### 3.2 Some Results for Specific Hypotheses classes

*Claim.* For every combinatorial optimization maximization problem  $\mathcal{P} = (\mathcal{I}, \mathcal{T}, \Pi)$  with a metric  $d$  over the space of its solutions, and for every solution  $I \in \mathcal{T}$  and  $\rho > 0$ ,

$$\sup_{T \in \mathcal{T}} \inf_{T' \in B(T, \rho)} \Pi(I, T') \geq \sup_{T \in \mathcal{T}} \Pi(I, T)(1 - \rho \lambda_{(I, T)}^\rho)$$

**Theorem 3.** *For every  $\rho > 0$ , the Best Separating Hyper-plane problem (for both the general and homogenous hyper-planes), as well as the Densest Open Ball Problem, have a polynomial time  $\rho$ -approximation algorithms that find solutions satisfying*

$$\Pi(I, A(I)) \geq \sup_{T \in \mathcal{T}} \inf_{T' \in B(T, \rho)} \Pi(I, T')$$

**Corollary 1.** *For each of the problems BSH, BSHH and DOP, for every  $\rho > 0$ , there exist a polynomial time  $\rho$ -approximation algorithm.*

## 4 Discussion

The work reported here is the beginning of an ambitious project aimed to identify which features of real-life inputs make them easier to process than the complexity predictions of worst-case analysis. Once such features are detected, we wish to design measures of complexity that will take these features into account and result in realistic computational complexity estimates.

The current paper focuses on one such feature. We believe that natural inputs for learning problems (that is, application-generated training data) are robust, in the sense that small perturbations of the parameters of learnt hypotheses should not result in dramatic changes in the training errors. To reflect this belief, this paper puts forward a new measure of the quality of approximations to optimization problems. This measure reduces to common approximation measures for robust inputs, while being more lenient on non-robust inputs. We prove, in Theorem 3 and Corollary 1, that there exist polynomial time algorithms that are guaranteed to find arbitrarily good approximations (in the new sense) to the optimization problems resulting from basic some learning tasks. Assuming that it is indeed the case that real-life data is robust, our results demonstrate that the new measure of complexity overcomes (at least some of) the over-pessimism phenomena of worst-case complexity.

## References

1. Levin, L.A.: Robust measures of information. *Comput. J.* **42** (1999) 284–286
2. Ben-David, S., Chor, B., Goldreich, O., Luby, M.: On the theory of average case complexity. *J. Comput. Syst. Sci.* **44** (1992) 193–219
3. Fellows, M.R.: Parameterized complexity: The main ideas and connections to practical computing. *Electr. Notes Theor. Comput. Sci.* **61** (2002)

4. Spielman, D.A., Teng, S.H.: Smoothed analysis of algorithms: Why the simplex algorithm usually takes polynomial time. *J. ACM* **51** (2004) 385–463
5. Ben-David, S., Eiron, N., Long, P.M.: On the difficulty of approximately maximizing agreements. *J. Comput. Syst. Sci.* **66** (2003) 496–514
6. Ben-David, S., Eiron, N., Simon, H.U.: The computational complexity of densest region detection. *J. Comput. Syst. Sci.* **64** (2002) 22–47