Function Evaluation: decision trees optimizing simultaneously worst and expected testing cost

Ferdinando Cicalese
Dept. Comp. Sc., University of Salerno, Italy

Eduardo Laber and **Aline Medeiros Saettler**
Dept. Comp. Sc., PUC-Rio, Brazil

Abstract

In several applications of automatic diagnosis and active learning a central problem is the evaluation of a discrete function by adaptively querying the values of its variables until the values read uniquely determine the value of the function. In general reading the value of a variable is done at the expense of some cost (computational or possibly a fee to pay the corresponding experiment). The goal is to design a strategy for evaluating the function incurring little cost (in the worst case or in expectation according to a prior distribution on the possible variables' assignments).

Our algorithm builds a strategy (decision tree) which attains a logarithmic approximation simultaneously for the expected and worst cost spent. This is best possible since, under standard complexity assumption, no algorithm that can guarantee $o(\log n)$ approximation.

Introduction

An automatic agent that implements a high frequency trading strategy decides the next action to be performed as sending or canceling a buy/sell order, on the basis of some market variables as well as private variables (e.g., stock price, traded volume, volatility, order books distributions as well as complex relations among these variables). For instance in (Nevmyvaka, Feng, and Kearns 2006) the trading strategy is learned in the form of a discrete function, described as a table, that has to be evaluated whenever a new scenario is faced and an action (sell/buy) has to be taken. The evaluation can be performed by computing every time the value of each single variable. However, this might be very expensive. By taking into account the structure of the function together with information on the probability distribution on the states of the market and also the fact that some variables are more expensive (or time consuming) to calculate than others, the algorithm can significantly speed up the evaluation of the function. Since market conditions change on a millisecond basis, being able to react very quickly to a new scenario is the key to a profitable strategy.

In a classical Bayesian active learning problem, the task is to select the right hypothesis from a possibly very large set $\mathcal{H} = \{h_1, \ldots, h_n\}$. Each $h \in \mathcal{H}$ is a mapping from a set $\mathcal{X}$ called the query/test space to the set (of labels) $\{1, \ldots, \ell\}$. It is assumed that the functions in $\mathcal{H}$ are unique, i.e., for each pair of them there is at least one point in $\mathcal{X}$ where they differ. There is one function $h^* \in \mathcal{H}$ which provides the correct labeling of the space $\mathcal{X}$ and the task is to identify it through queries/tests. A query/test coincides with an element $x \in \mathcal{X}$ and the result is the value $h^*(x)$. Each test x has an associated cost $c(x)$ that must be paid in order to acquire the response $h^*(x)$, since the process of labeling an example may be expensive either in terms of time or money (e.g. annotating a document). The goal is to identify the correct hypothesis spending as little as possible.

In an example of automatic diagnosis, $\mathcal{H}$ represents the set of possible hypotheses and $\mathcal{X}$ the set of symptoms or medical tests, with h^* being the exact diagnosis that has to be achieved by reducing the cost of the examinations. In (Bellala, Bhavnani, and Scott 2012), a more general variant of the problem was considered where rather than the diagnosis it is important to identify the therapy (e.g., for cases of poisoning it is important to quickly understand which antidote to administer rather than identifying the exact poisoning). This problem can be modeled by defining a partition $\mathcal{P}$ on $\mathcal{H}$ with each class of $\mathcal{P}$ representing the subset of diagnoses which requires the same therapy. The problem is then how to identify the class of the exact h^* rather than h^* itself. This model has also been studied in (Golovin, Krause, and Ray 2010) to tackle the problem of erroneous tests' responses in Bayesian active learning.

All problems above can be cast into the following.

The Discrete Function Evaluation Problem (DFEP). An instance of the problem is defined by a quintuple $(S, C, T, \mathbf{p}, \mathbf{c})$, where $S = \{s_1, \ldots, s_n\}$ is a set of objects, $C = \{C_1, \ldots, C_m\}$ is a partition of S into m classes, T is a set of tests, $\mathbf{p}$ is a probability distribution on S, and $\mathbf{c}$ is a cost function assigning to each test t a cost $c(t) \in \mathbb{Q}^+$. A test $t \in T$, when applied to an object $s \in S$, incurs a cost $c(t)$ and outputs a number $t(s)$ in the set $\{1, \ldots, \ell\}$. It is assumed that the set of tests is complete, in the sense that for any distinct $s_1, s_2 \in S$ there exists a test t such that $t(s_1) \neq t(s_2)$. The goal is to define a testing procedure which uses tests from T and minimizes the testing cost (in expectation and/or in the worst case) for identifying the class of an unknown object s^* chosen according to distribution $\mathbf{p}$.

The DFEP can be rephrased in terms of minimizing the cost of evaluating a discrete function that maps points (corresponding to objects) from some finite subset of $\{1, \ldots, \ell\}^{|T|}$ into values (corresponding to classes), where an object

Object	t1	t2	t3	Class	Probability
1	1	1	2	A	0.1
2	1	2	1	A	0.2
3	2	2	1	B	0.4
4	1	2	2	C	0.25
5	2	2	2	C	0.05

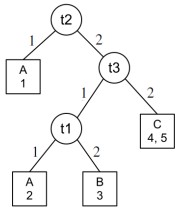

Figure 1: Decision tree for 5 objects presented in the table in the left, with $c(t1) = 2$, $c(t2) = 1$ and $c(t3) = 3$. Letters and numbers in the leaves indicate, respectively, classes and objects.

$s \in S$ corresponds to the point $(t_1(s), \ldots, t_{|T|}(s))$ obtained by applying each test of T to s. This perspective motivates the name we chose for the problem. However, for the sake of uniformity with more recent work (Golovin, Krause, and Ray 2010; Bellala, Bhavnani, and Scott 2012) we employ the definition of the problem in terms of objects/tests/classes.

Decision Tree Optimization. Any testing procedure can be represented by a *decision tree*, which is a tree where every internal node is associated with a test and every leaf is associated with a set of objects that belong to the same class. More formally, a decision tree D for $(S, C, T, \mathbf{p}, \mathbf{c})$ is a leaf associated with class i if every object of S belongs to the same class i. Otherwise, the root r of D is associated with some test $t \in T$ and the children of r are decision trees for the sets $\{S_t^1, ..., S_t^\ell\}$, where S_t^i, for $i = 1, \ldots, \ell$, is the subset of S that outputs i for test t.

Given a decision tree D, rooted at r, we can identify the class of an unknown object s^* by following a path from r to a leaf as follows: first, we ask for the result of the test associated with r when performed on s^*; then, we follow the branch of r associated with the result of the test to reach a child r_i of r; next, we apply the same steps recursively for the decision tree rooted at r_i. The procedure ends when a leaf is reached, which determines the class of s^*.

We define $cost(D, s)$ as the sum of the tests' cost on the root-to-leaf path from the root of D to the leaf associated with object s. Then, the *worst testing cost* and the *expected testing cost* of D are, respectively, defined as

$$cost_W(D) = \max_{s \in S}\{cost(D, s)\} \quad (1)$$

$$cost_E(D) = \sum_{s \in S} cost(D, s)p(s) \quad (2)$$

Figure 1 shows an instance of the DFEP and a decision tree for it. The tree has worst testing cost $1 + 3 + 2 = 6$ and expected testing cost $(1 \times 0.1) + (6 \times 0.2) + (6 \times 0.4) + (4 \times 0.3) = 4.9$.

Our Results

Our main result is an algorithm that builds a decision tree whose expected testing cost and worst testing cost are at most $O(\log n)$ times the minimum possible expected testing cost and the minimum possible worst testing cost, respectively. In other words, the decision tree built by our algorithm achieves simultaneously the best possible approximation achievable with respect to both the expected testing cost and the worst testing cost. In fact, for the special case where each object defines a distinct class—known as the *identification problem*— both the minimization of the expected testing cost and the minimization of the worst testing cost do not admit a sub-logarithmic approximation unless $\mathcal{P} = \mathcal{NP}$, as shown in (Chakaravarthy et al. 2007) and in (Laber and Nogueira 2004), respectively. In fact, it can be shown that the same inapproximability results hold in general for the case of m classes for any $m \geq 2$.

It should be noted that in general there are instances for which the decision tree that minimizes the expected testing cost has worst testing cost much larger than that achieved by the decision tree with minimum worst testing cost. Also there are instances where the converse happens. Therefore, it is reasonable to ask whether it is possible to construct decision trees that are efficient with respect to both performance criteria. This might be important in practical applications where only an estimate of the probability distribution is available which is not very accurate. Also, in scenarios like the one depicted in (Bellala, Bhavnani, and Scott 2012), very high cost (or time) might have disastrous consequences. In such a case, one could try to minimize the expected testing cost with the additional constraint that the worst testing cost shall not be much larger than the optimal one.

With respect to the minimization of the expected testing cost, our result improves upon the previous $O(\log 1/p_{\min})$ approximation shown in (Golovin, Krause, and Ray 2010) and (Bellala, Bhavnani, and Scott 2012), where p_{min} is the minimum positive probability among the objects in S. From the result in these papers an $O(\log n)$ approximation could be attained only for the particular case of uniform costs via a technique used in (Kosaraju, Przytycka, and Borgstrom 1999).

From a high-level perspective, our method closely follows the one used by (Gupta, Nagarajan, and Ravi 2010) for obtaining the $O(\log n)$ approximation for the expected testing cost in the identification problem. Both constructions of the decision tree consist of building a path (backbone) that splits the input instance into smaller ones, for which decision trees are recursively constructed and attached as children of the nodes in the path. However, our algorithm is more transparently linked to the structure of the problem, which remained somehow hidden in (Gupta, Nagarajan, and Ravi 2010) where the result was obtained via an involved mapping from adaptive TSP. Moreover, our algorithm avoids expensive computational steps as the Sviridenko procedure (Sviridenko 2004) and some non-intuitive/redundant steps that are used to select the tests for the backbone of the tree. In fact, we believe that providing an algorithm that is much simpler to implement and an alternative proof of the result in (Gupta, Nagarajan, and Ravi 2010) is an additional contribution of this paper.

State of the art

The DFEP has been recently studied under the names of class equivalence problem (Golovin, Krause, and Ray 2010) and group identification problem (Bellala, Bhavnani, and Scott 2012) and long before it had been described in the excellent survey by Moret (Moret 1982). Both (Golovin, Krause, and Ray 2010) and (Bellala, Bhavnani, and Scott 2012) give $O(\log(1/p_{min}))$ approximation algorithms for the version of the DFEP where the exptected testing cost has to be minimized. When the testing costs are uniform both algorithms can be converted into a $O(\log n)$ approximation algorithm via the approach of (Kosaraju, Przytycka, and Borgstrom 1999). The minimization of the worst testing cost—for which our algorithm provides the best possible approximation—had been left as an open problem in (Bellala, Bhavnani, and Scott 2012).

The variant of the DFEP where each object belongs to a different class—known as the *identification problem*—has been more extensively investigated (Dasgupta 2004; Adler and Heeringa 2008; Chakaravarthy et al. 2007; 2009). Both the minimization of the worst and the expected testing cost do not admit a sublogarithmic approximation unless $\mathcal{P} = \mathcal{NP}$ as proved by (Laber and Nogueira 2004) and (Chakaravarthy et al. 2007). For the expected testing cost, in the variant with multiway tests, non uniform probabilities and non uniform testing costs, an $O(\log(1/p_{min}))$ approximation is given in (Guillory and Bilmes 2009). In (Gupta, Nagarajan, and Ravi 2010) this result is improved to $O(\log n)$ employing new techniques not relying on the Generalized Binary Search (GBS)—the basis of all the previous strategies.

An $O(\log n)$ approximation algorithm for the minimization of the worst testing cost for the identification problem has been given by (Arkin et al. 1993) for binary tests and uniform cost and in (Hanneke 2006) for case with mutiway tests and non-uniform testing costs. The worst case cost is also investigated in (Guillory and Bilmes 2011) under the framework of covering and learning.

Other versions of the DFEP like the evaluation of AND/OR trees (a.k.a. read-once formulas) and the evaluation of Game Trees (a central task in the design of game procedures) are discussed in (Tarsi 1983; Saks and Wigderson 1986; Greiner et al. 2005). Very recently, approximation algorithms for stochastic Boolean function evaluation were provided in in (Deshpande, Hellerstein, and Kletenik 2014). In (Charikar et al. 2002), discrete function evaluation is considered from the perspective of competitive analysis; for the case of Boolean functions, results in this alternative setting are also given in (Kaplan, Kushilevitz, and Mansour 2005; Cicalese and Laber 2011).

References

Adler, M., and Heeringa, B. 2008. Approximating optimal binary decision trees. APPROX '08 / RANDOM '08, 1–9.

Allen, S.; Hellerstein, L.; Kletenik, D.; and Ünlüyurt, T. 2013. Evaluation of DNF formulas. http://arxiv.org/abs/1310.3673.

Arkin, E. M.; Meijer, H.; Mitchell, J. S. B.; Rappaport, D.; and Skiena, S. S. 1993. Decision trees for geometric models. In *Proceedings of the ninth annual symposium on Computational geometry*, SCG '93, 369–378.

Bellala, G.; Bhavnani, S. K.; and Scott, C. 2012. Group-based active query selection for rapid diagnosis in time-critical situations. *IEEE Trans. Inf. Theor.* 58(1):459–478.

Chakaravarthy, V. T.; Pandit, V.; Roy, S.; Awasthi, P.; and Mohania, M. 2007. Decision trees for entity identification: approximation algorithms and hardness results. In *Proceedings of the twenty-sixth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, PODS '07, 53–62.

Chakaravarthy, V. T.; Pandit, V.; Roy, S.; and Sabharwal, Y. 2009. Approximating decision trees with multiway branches. In *Proceedings of the 36th International Colloquium on Automata, Languages and Programming: Part I*, ICALP '09, 210–221.

Charikar, M.; Fagin, R.; Guruswami, V.; Kleinberg, J. M.; Raghavan, P.; and Sahai, A. 2002. Query strategies for priced information. *Journal of Computer and System Sciences* 64(4):785–819.

Cicalese, F., and Laber, E. S. 2011. On the competitive ratio of evaluating priced functions. *J. ACM* 58(3):9.

Dasgupta, S. 2004. Analysis of a greedy active learning strategy. In *Advances in Neural Information Processing Systems 17*, 337–344.

Deshpande, A.; Hellerstein, L.; and Kletenik, D. 2014. Approximation algorithms for stochastic boolean function evaluation and stochastic submodular set cover. In *SODA*.

Golovin, D.; Krause, A.; and Ray, D. 2010. Near-optimal bayesian active learning with noisy observations. In Lafferty, J. D.; Williams, C. K. I.; Shawe-Taylor, J.; Zemel, R. S.; and Culotta, A., eds., *NIPS*, 766–774. Curran Associates, Inc.

Greiner, R.; Howard, R.; Jankowska, M.; and Malloy, M. 2005. Finding optimal satisficing strategies for and-or trees. *Artificial Intelligence* 170(1):19–58.

Guillory, A., and Bilmes, J. 2009. Average-case active learning with costs. In *Proceedings of the 20th international conference on Algorithmic learning theory*, ALT'09, 141–155.

Guillory, A., and Bilmes, J. 2011. Simultaneous learning and covering with adversarial noise. In Getoor, L., and Scheffer, T., eds., *ICML*, 369–376. Omnipress.

Gupta, A.; Nagarajan, V.; and Ravi, R. 2010. Approximation algorithms for optimal decision trees and adaptive tsp problems. In *Proceedings of the 37th international colloquium conference on Automata, languages and programming*, ICALP'10, 690–701.

Hanneke, S. 2006. The cost complexity of interactive learning. http://www.stat.cmu.edu/ shanneke/docs/2006/cost-complexity-working-notes.pdf.

Kaplan, H.; Kushilevitz, E.; and Mansour, Y. 2005. Learning with attribute costs. In *STOC*, 356–365.

Kosaraju, S. R.; Przytycka, T. M.; and Borgstrom, R. S. 1999. On an optimal split tree problem. In *Proceedings*

of the 6th International Workshop on Algorithms and Data Structures, WADS '99, 157–168. London, UK: Springer-Verlag.

Laber, E. S., and Nogueira, L. T. 2004. On the hardness of the minimum height decision tree problem. *Discrete Appl. Math.* 144:209–212.

Moret, B. M. E. 1982. Decision Trees and Diagrams. *ACM Computing Surveys* 593–623.

Nevmyvaka, Y.; Feng, Y.; and Kearns, M. 2006. Reinforcement learning for optimized trade execution. In Cohen, W. W., and Moore, A., eds., *ICML*, volume 148 of *ACM International Conference Proceeding Series*, 673–680. ACM.

Saks, M. E., and Wigderson, A. 1986. Probabilistic boolean decision trees and the complexity of evaluating game trees. In *FOCS*, 29–38.

Sviridenko, M. 2004. A note on maximizing a submodular set function subject to a knapsack constraint. *Operations Research Letters* 32(1):41 – 43.

Tarsi, M. 1983. Optimal search on some game trees. *Journal of the ACM* 30(3):389–396.

Wolsey, L. 1982. Maximising real-valued submodular functions. *Mathematics of Operation Research* 7(3):410–425.