Testing (Subclasses of) Halfspaces

Kevin Matulef[1], Ryan O'Donnell[2], Ronitt Rubinfeld[3], and Rocco Servedio[4]

[1] ITCS, Tsinghua University
matulef@csail.mit.edu
[2] Carnegie Mellon University
odonnell@cs.cmu.edu
[3] Massachusetts Institute of Technology
ronitt@csail.mit.edu
[4] Columbia University
rocco@cs.columbia.edu

Abstract. We address the problem of testing whether a Boolean-valued function f is a halfspace, i.e. a function of the form $f(x) = \mathrm{sgn}(w \cdot x - \theta)$. We consider halfspaces over the continuous domain $\mathbf{R}^n$ (endowed with the standard multivariate Gaussian distribution) as well as halfspaces over the Boolean cube $\{-1,1\}^n$ (endowed with the uniform distribution). In both cases we give an algorithm that distinguishes halfspaces from functions that are ϵ-far from any halfspace using only $\mathrm{poly}(\frac{1}{\epsilon})$ queries, independent of the dimension n.
In contrast to the case of general halfspaces, we show that testing natural subclasses of halfspaces can be markedly harder; for the class of $\{-1,1\}$-weight halfspaces, we show that a tester must make at least $\Omega(\log n)$ queries. We complement this lower bound with an upper bound showing that $O(\sqrt{n})$ queries suffice.

Keywords: halfspaces, linear thresholds functions

This article presents a summary of the results found in [13] and [12] regarding the testability of halfspaces and certain subclasses of halfspaces.

1 Introduction

A *halfspace* is a function of the form $f(x) = \mathrm{sgn}(w_1x_1 + \cdots + w_nx_n - \theta)$ where $w_1, ..., w_n, \theta \in \mathbf{R}$. The w_i's are called "weights," and θ is called the "threshold." The sgn function is 1 on arguments ≥ 0, and -1 otherwise. The inputs to f can be either Boolean or real. Here we will mainly be concerned with functions over the Boolean cube, i.e. functions of the form $f : \{-1,1\}^n \to \{-1,1\}$. Halfspaces are also known as *threshold functions* or *linear threshold functions*; for brevity we shall refer to them here as LTFs.

LTFs are a simple yet powerful class of functions, which for decades have played an important role complexity theory, optimization, and perhaps especially machine learning (see e.g. [9, 18, 2, 15, 14, 17]). A lot of attention has been paid

to the problem of learning LTFs- that is, given examples labeled according to an unknown LTF (either random examples or queries to the function), find an LTF that it is ϵ-close to. However, the question we want to address is that of *testing* LTFs. That is, given query access to a function, we would like to distinguish whether it is an LTF or whether it is ϵ-far from any LTF. Any proper learning algorithm can be used as a testing algorithm (see, e.g., the observations of [8]), but testing potentially requires fewer queries. Indeed, in situations where query access is available, a query-efficient testing algorithm can be used to check whether a function is close to an an LTF, before bothering to run a more intensive algorithm to learn which LTF it is close to.

2 LTFs are testable with poly$(1/\epsilon)$ queries

The main result in [13] is to show that halfspaces can be tested with a number of queries that is *independent* of n. In fact the dependence is only polynomial in $1/\epsilon$. We note that any learning algorithm — even one with black-box query access to f — must make at least $\Omega(\frac{n}{\epsilon})$ queries to learn an unknown LTF to accuracy ϵ under the uniform distribution on $\{-1,1\}^n$ (this follows easily from, e.g., the results of [11]). So at least in terms of relationship to n, our testing algorithm is a significant improvement over using a learning algorithm. More formally, our main result is the following:

Theorem 1 ([13]). *Let f be a Boolean function $f : \{-1,1\}^n \to \{-1,1\}$, and (as is standard in property testing) we measure the distance between functions with respect to the uniform distribution over $\{-1,1\}^n$. Then there is an algorithm with 2-sided error making* poly$(\frac{1}{\epsilon})$ *queries that accepts f with high probability if it is an LTF, and rejects with high probability if it ϵ-far from all LTFs.*

We remark that the class of halfspaces is qualitatively much different than the other classes of Boolean functions that we know how to test. Some previous works have used the method of "implicit learning" to test classes such as s-term DNF formulas and size-s decision trees [4]. However the implicit learning technique only works for classes of functions whose members are close to juntas. This is not the case here, since the class of halfspaces contains, for example, the majority function, which is not at all close to a junta. Other previous works have shown how to test classes with some algebraic structure, like parity functions and low-degree polynomials, but these classes also are quite different from halfspaces.

Characterizations and Techniques.

To prove our results, we establish new structural results about LTFs which essentially characterize them in terms of their degree-0 and degree-1 Fourier coefficients. For functions mapping $\{-1,1\}^n$ to $\{-1,1\}$ it has long been known [3] that any linear threshold function f is *completely specified* by the $n+1$ parameters consisting of its degree-0 and degree-1 Fourier coefficients (also referred to as its *Chow parameters*). While this specification has been used to *learn* LTFs in

various contexts [1, 7, 16], it is not clear how it can be used to construct efficient *testers* (for one thing this specification involves $n+1$ parameters, and we want a query complexity independent of n). Intuitively, we get around this difficulty by giving new characterizations of LTFs as those functions that satisfy a particular relationship between just *two* parameters, namely the degree-0 Fourier coefficient and the sum of the squared degree-1 Fourier coefficients. Moreover, our characterizations are robust in that if a function approximately satisfies the relationship, then it must be close to an LTF. This is what makes the characterizations useful for testing.

We first consider functions mapping $\mathbf{R}^n$ to $\{-1, 1\}$ where we view $\mathbf{R}^n$ as endowed with the standard n-dimensional Gaussian distribution. Our characterization is particularly clean in this setting and illustrates the essential approach that also underlies the much more involved Boolean case. On one hand, it is not hard to show that for every LTF f, the sum of the squares of the degree-1 Hermite coefficients[5] of f is equal to a particular function of $\mathbf{E}[f]$ — regardless of *which* LTF f is (we call this function W; it is essentially the square of the "Gaussian isoperimetric" function).

Conversely, we show that if $f : \mathbf{R}^n \to \{-1, 1\}$ is *any* function for which the sum of the squares of the degree-1 Hermite coefficients is within $\pm\epsilon^3$ of $W(\mathbf{E}[f])$, then f must be $O(\epsilon)$-close to an LTF — in fact to an LTF whose n weights are the n degree-1 Hermite coefficients of f. The value $\mathbf{E}[f]$ can clearly be estimated by sampling, and moreover it can be shown that a simple approach of sampling f on pairs of correlated inputs can be used to obtain an accurate estimate of the sum of the squares of the degree-1 Hermite coefficients. We thus obtain a simple and efficient test for LTFs under the Gaussian distribution.

To handle general LTFs over $\{-1, 1\}^n$, we first develop an analogous characterization and testing algorithm for the class of *balanced regular* LTFs over $\{-1, 1\}^n$; these are LTFs with $\mathbf{E}[f] = 0$ for which all degree-1 Fourier coefficients are small. The heart of this characterization is a pair of results which give Boolean-cube analogues of our characterization of Gaussian LTFs. We show that the sum of the squares of the degree-1 Fourier coefficients of any balanced regular LTF is approximately $W(0) = \frac{2}{\pi}$. Conversely, we show that any function f whose degree-1 Fourier coefficients are all small and whose squares sum to roughly $\frac{2}{\pi}$ is in fact close to an LTF — in fact, to one whose weights are the degree-1 Fourier coefficients of f. Similar to the Gaussian setting, we can estimate $\mathbf{E}[f]$ by uniform sampling and can estimate the sum of squares of degree-1 Fourier coefficients by sampling f on pairs of correlated inputs. (An additional algorithmic step is also required here, namely checking that all the degree-1 Fourier coefficients of f are indeed small; it turns out that this can be done by estimating the sum of *fourth* powers of the degree-1 Fourier coefficients, which can again be obtained by sampling f on (4-tuples of) correlated inputs.)

The general case of testing arbitrary LTFs over $\{-1, 1\}^n$ is substantially more complex. Very roughly speaking, the algorithm has three main conceptual steps:

[5] These are analogues of the Fourier coefficients for L^2 functions over $\mathbf{R}^n$ with respect to the Gaussian measure.

- First the algorithm implicitly identifies a set of $O(1)$ many variables that have "large" degree-1 Fourier coefficients. Even a single such variable cannot be explicitly identified using $o(\log n)$ queries; we perform the implicit identification using $O(1)$ queries by adapting an algorithmic technique from [6]. This is similar to the "implicit learning" approach in [4].
- Second, the algorithm analyzes the regular subfunctions that are obtained by restricting these implicitly identified variables; in particular, it checks that there is a single set of weights for the unrestricted variables such that the different restrictions can all be expressed as LTFs with these weights (but different thresholds) over the unrestricted variables. Roughly speaking, this is done using a generalized version of the regular LTF test that tests whether a *pair* of functions are close to LTFs over the same linear form but with different thresholds.
- Finally, the algorithm checks that there exists a single set of weights for the restricted variables that is compatible with the different biases of the different restricted functions. If this is the case then the overall function is close to the LTF obtained by combining these two sets of weights for the unrestricted and restricted variables. (Intuitively, since there are only $O(1)$ restricted variables there are only $O(1)$ possible sets of weights to check here.)

3 Testing a natural subclass of halfspaces requires more queries

Complementing the work in [13], in [12] we consider the problem of testing whether a function f belongs to a natural subclass of halfspaces, the class of ± 1-*weight halfspaces*. These are functions of the form $f(x) = \mathrm{sgn}(w_1x_1 + w_2x_2 + \cdots + w_nx_n)$ where the weights w_i all take values in $\{-1, 1\}$. Included in this class is the majority function on n variables, and all 2^n "reorientations" of majority, where some variables x_i are replaced by $-x_i$. Alternatively, this can be viewed as the subclass of halfspaces where all variables have the same amount of influence on the outcome of the function, but some variables get a "positive" vote while others get a "negative" vote.

For the problem of testing ± 1-weight halfspaces, we prove two main results:

1. **Lower Bound.** We show that any nonadaptive testing algorithm which distinguishes ± 1-weight halfspaces from functions that are ϵ-far from ± 1-weight halfspaces must make at least $\Omega(\log n)$ many queries. By a standard transformation (see e.g. [5]), this also implies an $\Omega(\log \log n)$ lower bound for adaptive algorithms. Taken together with [13], this shows that testing this natural subclass of halfspaces is more query-intensive then testing the general class of all halfspaces.
2. **Upper Bound.** We give a nonadaptive algorithm making $O(\sqrt{n} \cdot \mathrm{poly}(1/\epsilon))$ many queries to f, which outputs YES with probability at least $2/3$ if f is a ± 1-weight halfspace, and NO with probability at least $2/3$ if f is ϵ-far from any ± 1-weight halfspace.

We note that it follows from [11] that *learning* the class of ± 1-weight halfspaces requires $\Omega(n/\epsilon)$ queries. Thus, while some dependence on n is necessary for testing, our upper bound shows testing ± 1-weight halfspaces can still be done more efficiently than learning.

Although we prove our results specifically for the case of halfspaces with all weights ± 1, our methods can be used to obtain similar results for other subclasses of halfspaces such as $\{-1, 0, 1\}$-weight halfspaces (± 1-weight halfspaces where some variables are irrelevant).

Techniques. As is standard in property testing, our lower bound is proved using Yao's method. We define two distributions D_{YES} and D_{NO} over functions, where a draw from D_{YES} is a randomly chosen ± 1-weight halfspace and a draw from D_{NO} is a halfspace whose coefficients are drawn uniformly from $\{+1, -1, +\sqrt{3}, -\sqrt{3}\}$. We show that a random draw from D_{NO} is with high probability $\Omega(1)$-far from every ± 1-weight halfspace, but that any set of $o(\log n)$ query strings cannot distinguish between a draw from D_{YES} and a draw from D_{NO}.

Our upper bound is achieved by an algorithm which uniformly selects a small set of variables and checks, for each selected variable x_i, that the magnitude of the corresponding singleton Fourier coefficient $|\hat{f}(i)|$ is close to to the right value. We show that any function that passes this test with high probability must have its degree-1 Fourier coefficients very similar to those of some ± 1-weight halfspace, and that any function whose degree-1 Fourier coefficients have this property must be close to a ± 1-weight halfspace. At a high level this approach is similar to some of what is done in [13], but here we are estimating $\sum_i |\hat{f}(i)|$ rather than $\sum_i \hat{f}(i)^2$. In both instances we are checking that the contribution of the degree-1 Fourier coefficients is "large," but in the second case we are estimating the coefficients more accurately in order to insure to insure that we only pass functions close to ± 1-weight halfspaces.

4 Open questions

Several questions related to testing halfspaces are still open. Here we point out a just a few:

- First is the question of whether there is a simpler algorithm for testing the general class of halfspaces over the Boolean cube. Although our algorithm makes "only" $\mathrm{poly}(1/\epsilon)$ queries, the exponent of the polynomial is something like 4000. Our algorithm is quite complicated, and hardly seems optimal. Obviously a more efficient algorithm utilizing new ideas would be preferred.
- Our current approach to testing halfspaces makes two-sided error. It is unclear whether this is necessary. In order to get a better handle on testing halfspaces, we might restrict ourselves to the question of one-sided testing. Can we devise a one-sided tester, or show that there is none? We conjecture (albeit without much confidence) that one-sided testing requires a query

complexity dependent on n. We make this conjecture based on the fact that for any constant k, there exist boolean functions which are not halfspaces, yet are consistent with a halfspace on any set of less than k examples [10].
– Perhaps the most obvious lingering question is whether we can extend our algorithm for LTFs to test degree-d polynomial threshold functions, or PTFs. This seems to require a significant amount of extra machinery, for example in relating the size of the degree-d Fourier coefficients to the weights of the corresponding terms inside a PTF, and to the bias of the PTF. Although there are some highly technical obstacles, given all of the recent structural results on PTFs, there is some hope that a testing algorithm can be achieved.

Acknowledgments

K.M. was supported in part by the National Natural Science Foundation of China Grant 60553001, and the National Basic Research Program of China Grant 2007CB807900,2007CB807901.

References

1. A. Birkendorf, E. Dichterman, J. Jackson, N. Klasner, and H.U. Simon. On restricted-focus-of-attention learnability of Boolean functions. *Machine Learning*, 30:89–123, 1998.
2. H. Block. The Perceptron: a model for brain functioning. *Reviews of Modern Physics*, 34:123–135, 1962.
3. C.K. Chow. On the characterization of threshold functions. In *Proceedings of the Symposium on Switching Circuit Theory and Logical Design (FOCS)*, pages 34–38, 1961.
4. I. Diakonikolas, H. Lee, K. Matulef, K. Onak, R. Rubinfeld, R. Servedio, and A. Wan. Testing for concise representations. In *Proc. 48th Ann. Symposium on Computer Science (FOCS)*, pages 549–558, 2007.
5. E. Fischer. The art of uninformed decisions: A primer to property testing. *Bulletin of the European Association for Theoretical Computer Science*, 75:97–126, 2001.
6. E. Fischer, G. Kindler, D. Ron, S. Safra, and A. Samorodnitsky. Testing juntas. In *Proceedings of the 43rd IEEE Symposium on Foundations of Computer Science*, pages 103–112, 2002.
7. P. Goldberg. A Bound on the Precision Required to Estimate a Boolean Perceptron from its Average Satisfying Assignment. *SIAM Journal on Discrete Mathematics*, 20:328–343, 2006.
8. O. Goldreich, S. Goldwaser, and D. Ron. Property testing and its connection to learning and approximation. *Journal of the ACM*, 45:653–750, 1998.
9. A. Hajnal, W. Maass, P. Pudlak, M. Szegedy, and G. Turan. Threshold circuits of bounded depth. *Journal of Computer and System Sciences*, 46:129–154, 1993.
10. L. Hellerstein. On generalized constraints and certificates. *Discrete Mathematics*, 226(211-232), 2001.
11. S. Kulkarni, S. Mitter, and J. Tsitsiklis. Active learning using arbitrary binary valued queries. *Machine Learning*, 11:23–35, 1993.

12. K. Matulef, R. Rubinfeld, R. A. Servedio, and R. O'Donnell. Testing -1,1 weight halfspaces. In *13th International Workshop on Randomness and Computation (RANDOM)*, 2009.
13. Kevin Matulef, Ryan O'Donnell, Ronitt Rubinfeld, and Rocco A. Servedio. Testing halfspaces. In *20th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 256–264, 2009.
14. M. Minsky and S. Papert. *Perceptrons: an introduction to computational geometry*. MIT Press, Cambridge, MA, 1968.
15. A. Novikoff. On convergence proofs on perceptrons. In *Proceedings of the Symposium on Mathematical Theory of Automata*, volume XII, pages 615–622, 1962.
16. R. Servedio. Every linear threshold function has a low-weight approximator. *Computational Complexity*, 16(2):180–209, 2007.
17. J. Shawe-Taylor and N. Cristianini. *An introduction to support vector machines*. Cambridge University Press, 2000.
18. A. Yao. On ACC and threshold circuits. In *Proceedings of the Thirty-First Annual Symposium on Foundations of Computer Science*, pages 619–627, 1990.