RESEARCH STATEMENT

SAM HOPKINS

Introduction

The last decade has seen enormous improvements in the practice, prevalence, and usefulness of machine learning. Deep nets give our phones ears; matrix completion gives our televisions good taste. Data-driven artificial intelligence has become capable of the difficult—driving a car on a highway—and the nearly impossible—distinguishing cat pictures without human intervention [18].

This is the stuff of science fiction. But we are far behind in explaining why many algorithms—even now-commonplace ones for relatively elementary tasks—work so stunningly well as they do in the wild. In general, we can prove very little about the performance of these algorithms; in many cases provable performance guarantees for them appear to be beyond attack by our best proof techniques. This means that our current-best explanations for how such algorithms can be so successful resort eventually to (intelligent) guesswork and heuristic analysis. This is not only a problem for our intellectual honesty. As machine learning shows up in more and more mission-critical applications it is essential that we understand the guarantees and limits of our approaches with the certainty afforded only by rigorous mathematical analysis.

The theory-of-computing community is only beginning to equip itself with the tools to make rigorous attacks on learning problems. The sophisticated learning pipelines used in practice are built layer-by-layer from simpler fundamental statistical inference algorithms. We can address these statistical inference problems rigorously, but only by first landing on tractable problem definitions. In their traditional theory-of-computing formulations, many of these inference problems have intractable worst-case instances. This demands that we carve up old input spaces in new ways. We make distributional and deterministic assumptions on our input data to bypass brittle combinatorial hardness constructions. This is not just a matter of convenience: redefining our problems sheds light on the nature of the real-world data for which we know empirically that statistical inference is tractable.

There is just as much to be done once we find mathematical problems which more closely align with tractable real-world instances. We need new algorithms which exploit the structure of tractable learning problems in a way amenable to rigorous analysis. In turn, we must use these algorithms and analyses to explain algorithmic behavior in the wild and even to shape the direction of future practical algorithmic machine learning research.

I seek to understand the algorithmic mathematics underlying fundamental machine learning problems.

- *Which properties of random inputs make hard optimization problems become feasible?*
- *Which problems remain unsolved for lack of sufficient algorithmic technology, and which really lie beyond the limits of efficient average-case computation?*

My twofold response to these questions is: *(1)* to develop algorithms with provable run-time and statistical accuracy guarantees and *(2)* to prove lower bounds for average-case versions of these problems against our most powerful convex-relaxation-based algorithms.

Current Research

My work focuses on applications of the Sum of Squares (SoS) meta-algorithm to fundamental learning problems. The SoS approach is the most powerful of a family of convex-relaxation techniques originating in combinatorial optimization [17, 20, 19, 22]. It captures our best algorithms for an extraordinary array of hard classical optimization problems. Under the notorious unique games conjecture (and $\mathsf{P} \neq \mathsf{NP}$), a relatively weak version of SoS (in the spirit of the maximum-cut algorithm of Goemans and Williamson [8]) yields the best possible polynomial-time algorithm for any constraint satisfaction problem [15, 21]. It also underlies the best-known approximation algorithms for the sparsest cut and unique games problems [4, 5]. We therefore see the success or failure of the SoS approach for any problem as a litmus test for that problem's polynomial-time tractability, period.

Planted Cliques in Random Graphs. How difficult is it to find a large clique in a random graph? This question dates at least back to Karp in the 1970s [14]. In the modern formulation, due to Jerrum [13] and Kucera [16], a clique of size ω is added to (*planted* in) the Erdös-Renyi random graph $G(n, 1/2)$, and the algorithmic task is to find it. This is one of the most basic average-case problems, and it connects closely to more typical machine learning average case problems, such as Sparse Principal Component Analysis [6].

As soon as ω is large enough that the problem is well-defined—the ω-size clique will be the largest in the graph—there is an algorithm running in time $n^{O(\log n)}$. This is much less than the (believed) best possible for NP-complete problems, for which it is believed that no algorithms much faster than exponential-time should exist. For this reason, under the conjecture that this *quasi-polynomial* running time is the best possible, planted clique also gives a starting point to study other problems with complexity in between polynomial and exponential. For example, there are connections to the complexity finding approximate Nash equilibria [9], to finding the densest subgraphs in worst-case graphs [2], and *free games* in hardness of approximation [1].

The best known polynomial-time algorithm for planted clique works only when ω is an exponential factor larger. It is a simple spectral one, using the top eigenvector of the adjacency matrix [3]. Variants of this algorithm are used for a wide range of average-case recovery problems. If there turns out to be no polynomial-time algorithm which does better even in the restricted setting of planted clique, this would be strong evidence that for the wide class of problems related to planted clique, spectral algorithms are optimal among polynomial-time algorithms.

In the worst-case world there is a recipe to show that an algorithm is the best possible in polynomial-time: via a reduction, show that to do better would also involve solving NP-complete problems efficiently. But reductions among average-case problems tend to result in un-natural input distributions (which miss the point of the average case entirely), so their usefulness here is inherently limited. At present our best substitutes are concrete lower bounds for particular classes of algorithms. Since SoS captures nearly all of our best polynomial-time approximation algorithms, the frontier in hardness of planted clique is to show that no polynomial-time SoS algorithm improves on the performance of the simple spectral one.

My collaborators and I have improved the best-known impossibility results to show that a moderately strong version of SoS (the *degree-four* algorithm) does not substantially improve on the spectral algorithm [10]. We anticipate improving this result to show that no polynomial-time algorithm using the SoS approach can substantially beat the spectral algorithm. Previous results along these lines only rule out algorithms (weaker than the SoS approach) which can draw upon only very *local* properties in the input graph [7]. By contrast, our work is on understanding optimality certificates for the SoS approach, which can and does exploit nontrivial *global* statistical information about the graph.

It remains tantalizingly open (and a problem I will continue to attack) whether the simple spectral algorithm can be beaten using the SoS approach in just slightly super-polynomial running time.

Speeding up SoS Algorithms. Can the nominally-impractical SoS approach (in full generality it requires solving a large semidefinite program) yield practical algorithms for average case problems? My collaborators and I have demonstrated that for a number of machine learning problems where the SoS approach provides best-known provable guarantees, the *proof* that SoS approach succeeds can itself be exploited to give a fast spectral algorithm with similar guarantees (we have even run these algorithms on sizable instances) [11, 12].

We give a general recipe to turn a proof that the SoS approach works for a particular problem into into a fast spectral algorithm, so long as an approximate version of that proof can be constructed quickly from the input. Traditional spectral algorithms focus on a single matrix associated to the input—the adjacency matrix of a graph, for example. By contrast, SoS associates a large space of matrices to the data, and in full generality can find the best matrix in that space. The spectral algorithms we get from speeding up SoS still exploit this larger space of available matrices, so they can (provably) out-perform traditional spectral methods. (This is in contrast to the planted clique problem from above; here we start with problems where polynomial-time SoS algorithms do improve on simple spectral methods.)

This technique gives the fastest known (and sometimes the only known polynomial-time) algorithms with such strong guarantees to recover a sparse vector from a random subspace (the *planted sparse vector* problem), to de-noise a low-rank tensor (the *tensor principal component analysis* problem), and to find the components of a high-rank random 3-tensor (the *overcomplete tensor decomposition* problem) [11].

Important questions remain. Aside from quantitative improvements to our algorithms (on which we already have work in progress), and applications of the recipe to other machine learning problems, we still lack a general theory of what constitutes a good enough efficiently-constructible approximation to the SoS optimality certificate, and exactly when such certificates can be compressed enough to speed up the spectral algorithms. Such a theory would almost certainly shed new light on the boundary between polynomial-time and not for important machine learning problems (for example, dictionary learning and learning mixture models).

Future Directions

Unified and Provable Machine Learning. The SoS approach offers an opportunity to understand many hard machine learning problems and sophisticated algorithms in a common framework. Such unification is crucial to complete the picture for theoretical machine learning. It will reveal shared structure underlying many solvable problems (giving new provable algorithms along the way), and highlight which differences are fundamental rather than symptoms of an incomplete algorithmic picture. I will continue to investigate new SoS algorithms which push forward the state of the art for provable performance guarantees, and I will explore how preexisting algorithms—with and without provable guarantees—relate to SoS-based algorithms.

Connections to central Theory Problems. The new algorithms and proof strategies we devise for machine learning problems will shed new light on problems traditionally central to theoretical computer science. The SoS approach holds promise to resolve long-standing open problems in worst-case algorithms—in particular, the *unique games conjecture*—and there is hope that insights from studying its average-case behavior will be helpful along the way. I will investigate how to apply progress in understanding SoS from the machine learning point of view to worst-case algorithms, with the eventual goal of improved algorithms for worst-case sparse vector recovery problems, graph cut and expansion problems, and progress on the unique games conjecture.

REFERENCES

[1] Scott Aaronson, Russell Impagliazzo, and Dana Moshkovitz. Am with multiple merlins. In *Computational Complexity (CCC), 2014 IEEE 29th Conference on*, pages 44–55. IEEE, 2014.

[2] Noga Alon, Sanjeev Arora, Rajsekar Manokaran, Dana Moshkovitz, and Omri Weinstein. Inapproximability of densest κ-subgraph from average case hardness. *Unpublished manuscript*, 2011.

[3] Noga Alon, Michael Krivelevich, and Benny Sudakov. Finding a large hidden clique in a random graph. *Random Structures and Algorithms*, 13(3-4):457–466, 1998.

[4] Sanjeev Arora, Satish Rao, and Umesh V. Vazirani. Expander flows, geometric embeddings and graph partitioning. In *STOC*, pages 222–231, 2004.

[5] Boaz Barak, Prasad Raghavendra, and David Steurer. Rounding semidefinite programming hierarchies via global correlation. In *FOCS*, pages 472–481, 2011.

[6] Quentin Berthet and Philippe Rigollet. Complexity theoretic lower bounds for sparse principal component detection. In *Conference on Learning Theory*, pages 1046–1066, 2013.

[7] Uriel Feige and Robert Krauthgamer. The probable value of the lovász–schrijver relaxations for maximum independent set. *SIAM Journal on Computing*, 32(2):345–370, 2003.

[8] Michel X. Goemans and David P. Williamson. .879-approximation algorithms for max cut and max 2sat. In *STOC*, pages 422–431, 1994.

[9] Elad Hazan and Robert Krauthgamer. How hard is it to approximate the best nash equilibrium? *SIAM Journal on Computing*, 40(1):79–91, 2011.

[10] Samuel B. Hopkins, Pravesh Kothari, Aaron Potechin, Prasad Raghavendra, and Tselil Schramm. On the integrality gap of degree-4 sum of squares for planted clique. *SODA*, 2016.

[11] Samuel B. Hopkins, Tselil Schramm, Jonathan Shi, and David Steurer. Speeding up sum-of-squares for tensor decomposition and planted sparse vectors. *In Preparation*.

[12] Samuel B. Hopkins, Jonathan Shi, and David Steurer. Tensor principal component analysis via sums of squares. *COLT*, 2015.

[13] Mark Jerrum. Large cliques elude the metropolis process. *Random Struct. Algorithms*, 3(4):347–360, 1992.

[14] Richard M. Karp. Probabilistic analysis of some combinatorial search problems. *Algorithms and Complexity: New Directions and Recent Results*, 1976.

[15] Subhash Khot, Guy Kindler, Elchanan Mossel, and Ryan O'Donnell. Optimal inapproximability results for max-cut and other 2-variable csps? In *FOCS*, pages 146–154, 2004.

[16] Ludek Kucera. Expected complexity of graph partitioning problems. *Discrete Applied Mathematics*, 57(2-3):193–212, 1995.

[17] Jean B Lasserre. Global optimization with polynomials and the problem of moments. *SIAM Journal on Optimization*, 11(3):796–817, 2001.

[18] Quoc V. Le, Rajat Monga, Matthieu Devin, Greg Corrado, Kai Chen, Marc'Aurelio Ranzato, Jeffrey Dean, and Andrew Y. Ng. Building high-level features using large scale unsupervised learning. *CoRR*, abs/1112.6209, 2011.

[19] Yurii Nesterov. Squared functional systems and optimization problems. In *High performance optimization*, pages 405–440. Springer, 2000.

[20] Pablo A Parrilo. *Structured semidefinite programs and semialgebraic geometry methods in robustness and optimization*. PhD thesis, Citeseer, 2000.

[21] Prasad Raghavendra. Optimal algorithms and inapproximability results for every csp? In *STOC*, pages 245–254, 2008.

[22] NZ Shor. An approach to obtaining global extremums in polynomial mathematical programming problems. *Cybernetics*, 23(5):695–700, 1988.

CORNELL UNIVERSITY
E-mail address: `samhop@cs.cornell.edu`