MIT CSAIL Alliances | Sam Hopkins podcast

Welcome to MIT's Computer Science and Artificial Intelligence Labs Alliances podcast series. My name is Steve Lewis, I'm the Assistant Director of Global Strategic Alliances for CSAIL at MIT. In this podcast series, I will interview principal researchers at CSAIL to discover what they're working on and how it will impact society.

Sam Hopkins is a mathematician and computer scientist. He studies algorithms and computational complexity, especially, through the lens of convex programming. He is an assistant professor in the theory of computing group in the Department of Electrical Engineering and Computer Science at MIT. Sam joined CSAIL in 2022 after receiving a PhD at Cornell University and spending several years at UC Berkeley as a Miller fellow. Sam, thanks for your time today. Can you give our audience a rough definition of high dimensional statistics?

I think it helps to go back to thinking about the beginning of statistics in the first half of the 20th century just as a point of comparison. And statistics was developed to deal with data sets that were created by humans. Agricultural data sets or biological data sets or pharmaceutical data sets. I'm not a historian, so I can't go into detail about it. But the point is that in data sets like that, they can't be that big by definition. They're created by humans who have limited amounts of time and resources.

And every element of such a data set, like every crab that you're cataloging or every person in your drug trial, you collect a somewhat limited amount of information about them. You collect a few numbers. That's a low dimensional statistic setting, where every element in your data set consists of just a few numbers.

And the whole theory is, statistics was basically developed to deal with low-dimensional data. And in particular, an assumption that often gets made if you take STAT 101 is that the number of samples that you have in your data set, like the number of crabs that you cataloged goes to infinity while the number of numbers per element of the data set, like the pieces of data you're collecting per crab-- crab is a famous statistics example, is why it shows up in STAT 101 all the time, is why I'm using it. Stays constant.

High dimensional data means that the number of numbers you collect per data point, right? The number of parameters that you measure for each sample is also getting larger as the data set gets larger. That's how we think about it mathematically. But if you want to think about intuitively, think about the idea that, maybe, you're collecting 100 or 1,000 features or 10,000 features for every data point.

So that's much more common in modern computer or automatically generated data sets. And it doesn't have to be something fancy. Just think about pictures. Think about images. If you have a data set of images, then a, have a large number of images, but b, each image consists of a very large number of numbers. You have a number-- or several numbers, actually, for every pixel in the image.

And the mathematics of what you should do to extract useful information from a data set really fundamentally changes when you're in a setting that every individual sample consists of a number of numbers that is still pretty big compared to the number of samples that you have. So that's-- OK. That's high dimensional statistics in a sort of nutshell.

OK. Well, what kinds of decisions have to be made in high dimensional statistics when it comes to robustness? Or what is the robustness accuracy trade-off?

Robustness in statistics means, first, maybe let's get a working idea of what that means. So robustness means that whatever method you're using to process your data shouldn't be too affected by the possibility that there are, let's say, outliers in your data set. That your data set is not completely clean. Or maybe that the method that you've-- the assumptions that underlie the method that you're using to process your data are mildly violated by your data set.

When we develop statistical methods from a theoretical mathematical point of view, well, we like to have mathematical justifications for our methods, but those mathematical justifications typically require some kind of assumptions. A strong assumption would be, all your data came from a normal distribution or a Gaussian distribution. But there are weaker kinds of assumptions you could make to get some mathematics off the ground that would justify your statistics procedures.

And those assumptions are what underlie things like P values, confidence intervals. All the stuff that we like to see in scientific papers. What if those assumptions are violated? The real world doesn't necessarily respect even the most pessimistic mathematical assumptions. So we'd like to design statistical procedures for processing data that are robust to violations of our mathematical assumptions.

OK. So what's this trade off that I'm talking about? Well, of course, the worst the violation of an assumption is, the dirtier your data set, the more outliers it has, the less accuracy you'd expect from some kind of algorithm that's processing that data set to produce some conclusion. But you can make that question quantitative. As I vary the dirtiness of the data set, the amount of outliers, the amount of violation of an underlying mathematical assumption, how does the accuracy of what comes out the other side of my data processing pipeline change?

And what I'd like to do is design algorithms, methods for extracting useful information from a large noisy data set, which as you make the data set dirtier, their accuracy doesn't get too much worse. It doesn't get worse too fast. And you can study that trade off quantitatively. And you can ask-- well, you can say, well, maybe there's some fundamental limitation, there's some fundamental limit to the amount of accuracy that I can get based on having a data set which has a certain percentage of dirty data in it, of bad data.

And I'd like to design algorithms which match that fundamental limitation. The squeeze all that I can out of a data set even when it violates my underlying mathematical assumptions.

I see. So what is the relationship between privacy and robustness?

Yeah. So one thing I've been thinking about a lot over the last year or so is this question of private statistics. So this is going to sound very different right off the bat. So privacy is this idea that we often we're working with data sets that contain sensitive data where the data represents some features of individuals like people that those people might not like to have made public.

So for instance, maybe, a data set consists of sensitive medical records and you're trying to understand some genomic study or something like that. But the people in the data set don't want their whole genome published for everyone to see. And common situation, especially with modern machine learning models, is that they can release information about people in the underlying data set that they were trained on in surprising ways.

Even if it doesn't look like you're just publishing the data set, maybe there's some sophisticated way that you can use the output of a machine learning model to infer what you thought were secret private features of the underlying individuals in the data. So that issue justifies the study of private algorithms for high dimensional statistics.

So here private means that the output of that algorithm should, in some way, provably not leak information about individuals represented in the data set. Now, that's paradoxical, because the output has to somehow depend on the individuals in the data set. Because it's supposed to be doing statistics about those individuals. On the other hand, you don't want it to depend too heavily on any one individual.

So that if that individual were or were not in the data set, the output of that algorithm wouldn't change too much, let's say. And if you can have an algorithm with that property, we call this differential privacy, it provides some kind of plausible deniability to any individual who's in the data set. Because if somebody sees the output of this-- Some algorithm that's processing this data set, imagine that.

And it's good to think of a scientist conducting a medical study, and they've got a data set with a lot of sensitive information, but they're going to publish a paper at the end of the day. And that paper is going to be in the public domain. Somebody who participated in, say, the drug trial can plausibly deny their own participation.

Can say, oh, no. I wasn't in that data set. And nobody will be the wiser, because the way the published paper would have looked if they were not in the data set is not very different from the way that it looked with them in the data set. So that's this very important definition called differential privacy. And we could define all the squishy terms that I used mathematically, but we won't do it here.

And maybe you could hear in that definition that there is a similarity to this idea of robustness. And the similarity was that we wanted to be able to take a data set and change it, like add or remove an individual and not have the statistical conclusions that we drew change too much.

On the one hand, when we're talking about robustness, we were worried about that from an accuracy perspective. We thought, oh, the data set might be dirty and we want to be able to say that, well, even if there was some dirty data, or if I took out the dirty data or put in different dirty data, I wouldn't have changed my conclusions too much. That's robustness.

Privacy, we want the same property for a different reason. We want to protect the individuals who might or might not be in the data set. So this relationship, even though people understood that qualitatively, these two things seemed a bit similar already 15 years ago, it's only in the last couple of years that we've started to understand quantitatively how these two things are the same.

And remember this robustness accuracy trade off that I talked about before, well, there's also a privacy accuracy trade. The more privacy that I want where I could have made quantitative this relationship, I remove or add one individual to the data set, just how much do the conclusions that I drew from that data set change quantitatively.

You can study a privacy accuracy trade-off. And one thing we came to understand in the last year or two is that the robustness accuracy trade-off and the privacy accuracy trade-off on a mathematical level are, essentially, the same trade-off. So you can translate back and forth between these two.

And how similar or different is the private algorithms to homomorphic encryption? There's an important conceptual difference. So first of all, these ideas-- actually, these privacy ideas came out of the cryptography community. So it was people thinking about protecting secrets who arrived at these definitions.

But those is a really-- the definition of differential privacy, in particular. But there's a really important conceptual distinction. In cryptography broadly, fully homomorphic or not, the goal is to completely protect a secret. You don't want the public to learn anything about your computation or your communication or whatever it is that you're hiding.

In privacy, what we're thinking about is a situation where we're going to release some information to the public, because that's actually our goal. Imagine this scientific study setting. Our goal is to release a scientific paper to the public that says that this drug is effective or whatever conclusion we're drawing.

So by definition, the public is going to learn something. And that thing that the public is going to learn inherently reveals, at least, some information about the underlying individuals. Because if all of the individuals responded positively to the drug, then we would say the drug is effective. And if all of the individuals responded negatively to the drug, we would say the drug is ineffective.

So it releases some information. And in differential privacy, we're studying how can you how can you quantitatively control the amount of information you release about any particular individual while still getting good conclusions that reflect the broad statistics of an underlying data set? Does that make sense?

Yes.

So both of them are about protecting secrets, but in one case, you are trying to release something and mitigate the consequences. And in the other case, you're trying to release nothing.

So how does statistical inference create computational complexity?

So the relationship between statistics and computational complexity is a core theme in my research. So we've talked so far mostly about features of algorithms for high dimensional statistics. And when we talk about algorithms for high dimensional statistics or algorithms for anything, we usually want algorithms that run fast. That we can actually run on our data sets, on our computers.

And taking us all the way back to what we talked about at the start, the difference between low dimensional data and data sets created by humans and high dimensional data sets created often by computers. One difference is that when you're in the low dimensional small data setting, almost any computation that you'd like to do is going to be efficient enough, because the scale of the data you're working on is just not that large.

But in high dimensional large data settings, you can have a situation where the roadblock to extracting useful information from your data set is the amount of computation that you can throw at it instead of the amount of information present in your data. And we've seen this in practice over the last 10 years of improving-- ever improving machine learning models. We're continuing to learn that the bigger a deep net you throw at your image classification or your language generation task, the better you will get at it, even if your data set stays roughly fixed. We use this data set called ImageNet for 10 or 15 years.

The data stayed fixed, but we got bigger and bigger computational models attacking it. And we got better and better results. So the roadblock can be computation. And a big question addressed in my research is, when is that roadblock fundamental? Are there problems in high dimensional statistics where they're just-- where there is enough information in a data set, in a theoretical sense, to extract a useful conclusion from it.

But that no computationally efficient algorithm could ever do so. So how could that happen? Well, you could imagine a task where the conclusion that you want to draw requires you to conduct some kind of brute force search among an exponentially large number of possibilities. And a great example here where we think this happens is a problem called planted clique. Just to give you a more concrete idea. So this is a mathematical model of a problem that you might encounter in the real world.

Because it's a mathematical model, it's very simple and rarefied. But we use mathematical models to study much more complicated phenomena that you might see in practice. Here's the mathematical model. Imagine that what you have is a graph data set. So graph here means you have a bunch of nodes that are entities, and you have edges connecting those nodes. So pairwise relationships among entities.

And your goal is to find, in that graph, a large dense subgraph. So dense means that-- so a subgraph is a subcollection of entities, a set of nodes in the graph. And dense here means that there are a lot of edges among just that subset of notes. We can model this problem by simplifying it in the following way.

Imagine that the graph you're dealing with is just a random graph. Every pair of nodes in the graph is connected with probability one half. We just flip a coin for every single one. So this isn't a realistic statistics problem, it's a model to study computation. And we could ask to find, in that graph, a large so-called clique. A clique is a set of nodes where all of the pairs are connected. So we're modeling a dense subgraph as a graph where everything is connected.

And now, a naive way to go about this is just, if you want to say find a clique of size 10 in this graph, 10 nodes, all pairs are connected, start enumerating all sets of 10 vertices in the graph. 10 nodes in the graph right. If your graph has size n, then what you'll get is an n to the 10 time algorithm. And that is totally infeasible for any large-ish value of n.

What's interesting in this problem is that we believe-- we have very good reasons. Mathematical reasons to believe that actually, you can't beat that brute force algorithm for the problem of finding a large clique in a graph. There's a sense in which brute force should be the best that you can do. So finding large cliques in a random graph doesn't sound like a statistics problem, but it turns out that you can translate that problem into a lot of other problems that people think about.

There's the idea of doing reductions between statistics problems. So one example would be the so-called sparse principal component analysis problem. Some people who listen to your podcast might be familiar with principal component analysis. It's a really basic tool in processing large high dimensional data sets.

And a variant of it, that's actually people commonly want to do calls sparse principal component analysis, has-- we've talked about trade offs already in this podcast, we talked about robustness versus accuracy, we talked about privacy versus accuracy. Sparse PCA has a computation versus accuracy trade-off. And you can actually study that computation versus accuracy trade where the more computation you throw at it, the more accurate solution you can get.

But there's really a fundamental limit like-- there's a trade-off curve. And you can study that by studying these simple models like planted clique.

When we talk about the information computation gap, what is the information part of it?

Great. Great, great, great. So information computation gap is the term of art we use to describe a situation where there is enough information in a data set to draw a conclusion you want, but there's a gap between what you could do if you had infinite computational resources versus reasonable computational resources.

So this planted clique situation exactly exhibits and information computation gap. There is-- in a large random graph, there will be little highly connected subgraphs, little dense subgraphs. And you can imagine, in a graph data set, that actually a small highly connected subgraph is a useful thing to find. So you could consider drawing a useful conclusion from your data to find such a small subgraph.

A good example would be in a protein-protein interaction network. So you have a network data set where the entity is the nodes, are proteins and there are edges between a pair of proteins if they exhibit some kind of interaction together. And in such a network data set, a small highly connected subgraph can be a set of, imagine 10 proteins that all together participate in some useful chemical pathway in the cell.

So if you're a biologist trying to figure out what a bunch of proteins in the cell do, maybe, looking at those 10 together will give you some insight. But exactly this problem exhibits this gap. If you had n to the 10 time to spend, you could brute force over all possible sets of 10 nodes in your network, but that's just fundamentally infeasible.

So when you identify an information computation gap, well, first of all, there's all kinds of mathematics that goes into understanding, well, is this really an information computation gap or have we just not found a better algorithm? And a lot of people, including me, spend a lot of time thinking about how can you tell when you've really squeezed all of the algorithmic power? Like it's not just a matter of thinking a little longer, there really isn't a better algorithm here.

And knowing that is very useful, because if you know that you can't do better by just thinking about a better algorithm, now you have to go back and say, OK, how can we reformulate this problem? How can we bring additional data to bear on the task at hand so that we can break-- so we can close this gap? Because we know that more algorithm design isn't the way forward.

So what is the long term impact of this research? Where do you see the field of high dimensional statistics changing in the next 5, 10, 20 years?

I think, specifically, about algorithmic high dimensional statistics. What can you do when you have limited computational resources? And the field of algorithmic statistics, it's still in its infancy, and it's still an art in some sense. So these days, if you have some new high dimensional statistics problem, somebody working with a large data set encounters a new conclusion they'd like to draw out of this.

It's often a matter of an entire PhD's worth of research to understand what are the right algorithms for doing whatever task they want to do. And if you want to understand whether those algorithms are optimal, OK, you need to get another PhD student and really study the heck out of the limitations of those algorithms.

So by contrast, in mathematical areas that are well developed, we have general purpose tools that can take a new problem and very quickly tell you, OK, is this problem, what is the solution? Is this problem feasible? So a good example of this would be the area of combinatorial optimization, where we have this amazing theory of computational complexity called NP hardness-- the famous P versus NP problem factors in here.

And so in the area of combinatorial optimization, this is problems like traveling salesman and scheduling problems and things like this. We have all these mathematical tools that let you take a newly presented problem and pretty quickly figure out like in a day, OK, what's the optimal algorithm for this problem and give you some concrete evidence that doing better than that algorithm is just not possible. Just not mathematically possible.

So what we need-- and I think this is a 10 or 15-year project, is we need a comparable set of mathematical tools for algorithmic statistics that can take it from a thing where, if you're a practitioner in the field and you encounter a new problem, you need to go find a specialist, you need to go recruit PhD students just to figure out what the best algorithms are, to having a general purpose theory that is both-- it's predictive, right? It can take a new problem and predict for you. This is the best algorithm.

And explanatory. It can not only, say this is the best algorithm, it can tell you why this is the best one and why you can't do any better. And useful in the sense that it can tell you what additional data would you need to add in order to improve on the guarantees of the best possible algorithm. So that's where I'd love to be, but it's going to take us a long time.

And is this a machine learning model that needs to be developed in order to tell you this information?

That's a great question. I mean, we don't-- it's conceivable although, I think, that's related to the question of how much can machine learning do science? Like I'm talking about the same kind of simple rules that you find in physics. So I think that's a great question. I think it's a little bit orthogonal. There's two orthogonal questions here. One is, can machine learning do science?

Can you feed a bunch of data about the physical world into a machine learning model and have it come out with F equals ma? That's one question. Then another question is, what is the F equals ma OF high dimensional statistics? If we can use machine learning to get there, great. I don't know how to do it. But that would be awesome.

And what excites you about the field that's going on now or where you see things going in the future. First of all, in terms of broad impacts, maybe there's two I want to highlight. And then let me tell you, just mention I'll go back to a specific project I've been thinking about in the last year.

So one is that I think it's-- one feature of my research is working on new algorithms that just look fundamentally different from the current state of the art, say, machine learning or statistics methods. And I think time and time again, we see algorithmic paradigm shifts happen. And those can only happen if you have a bunch of crazy skunkworks algorithms ideas being developed that are, hopefully, 10 years ahead of whatever the state of the art is in practice.

Now, for any one such idea, yeah. There's a 1% or a 0.1% chance that it turns out to be the next generation of high dimensional statistics algorithmic breakthrough. But you need those kind of things. So a lot of my research in the field is studying computational problems and statistics where we don't know algorithms that can solve them and trying to design algorithms to solve them.

And the hope is that, in so doing, we discover new algorithmic insights. New general purpose algorithmic ideas that can then be applicable to a broad range of problems. And the current dominant paradigm in, say, machine learning, which is deep learning, has scalability issues.

It requires immense amounts of compute power to get state of the art performance on image classification or-- OK, these days people are super excited about large language models. But these things have huge, huge compute costs. So to me, that justifies and makes important and exciting questions of what are the trade offs between computational efficiency and algorithmic performance for accuracy at statistical tasks?

And hopefully, we will discover new algorithmic paradigms that improve on the performance versus compute cost efficiency of the current state of the art. So that's one hopeful long term impact. The flip side of that coin, the complexity side of that coin is that you need tools to understand when you've bottomed out. When you've sucked all of the algorithmic performance you can out of something.

So you need computational complexity on the other side to help guide you in your algorithm design. And right now, as I said, computational complexity for statistical inference is really a baby field. And we don't yet have very good tools to understand, when have you extracted all that you can when have you pushed the algorithms as far as they can go.

So those are broad things I hope will have impact in the lab in the next 15 years. Something that's exciting to me just that I've been working on in the last year or so is going back to this connection between privacy and robustness. So a common issue with this differentially private algorithms is that-- so we talked about the privacy versus accuracy trade-offs.

And for many applications that people want to do today, the privacy accuracy trade-off achieved by the best current private algorithms isn't good enough. There's just too much lost accuracy. And so people look for just-- people weaken the privacy guarantees, people will do things that are less good on the privacy front to get the accuracy that they need to do whatever task that they're doing.

I am pretty sure that even for very basic statistical tasks, think linear regression. Not fancy machine learning models, really basic workhorse stats. Even for those basic problems, we haven't hit the fundamental limits of the privacy-accuracy trade-off. There's still room for algorithmic improvements.

And I think we understand that in large part, because we understood what those fundamental limits were for robustness and we came to understand how to port the mathematical picture for the robustness accuracy trade-off over to the privacy world. So I've been thinking about, how can we close those gaps in private statistics? How can we design better algorithms, because even though I think there is some sentiment that privacy will-- differential privacy, anyway, will never work, it's the wrong idea because the current generation of algorithms has too much accuracy loss, I don't think that's fundamental.

I think there are better algorithms out there, and my students and I are working on designing better algorithms that can, actually, be implemented and run and we're going to start running tests on real data sets to see if we can improve on the accuracy guarantees while maintaining strong privacy guarantees.

And where could people go to find out more about your research? So maybe two places. If you wanted to read mathematical papers, you could, of course, go to my website where I have lots of them posted. You can also, if you go to my website, you'll see a link to a YouTube channel where I have a couple of general audience talks available for people to take a look at.

Great. And what is the URL for your website? Oh, it's SamuelBHopkins.com.

Sam, thank you very much for your time.

Thank you so much for having me. It's been a pleasure.

If you're interested in learning more about the CSAIL Alliance Program and the latest research at CSAIL, please visit our website at cap.csail.mit.edu. And listen to our podcast series on Spotify, Apple Music, or wherever you listen to your podcasts. Tune in next month for a brand new edition of the CSAIL Alliance's podcast, and stay ahead of the curve.