EPISODE 340

[INTRODUCTION]

[0:00:00.4] JM: Humans have now been defeated by computers at Heads-up No Limit Hold'em Poker. Some people thought that this wouldn't be possible. Sure, we can teach a computer to beat a human at Go, or Chess, those games have a smaller decision space. There's no hidden information, there's no bluffing. Poker, must be different. It's too human to be automated.

The game space of poker is different than that of Go. It has 10 to the 160 different situations which is more than the number of atoms in the universe and the game space keeps getting bigger as the stack sizes of the two competitors gets bigger. It's still possible for a computer to beat a human at calculating game theory optimal decisions if you approach the problem correctly, that is.

Lebradas was developed by CMU professor Tuomas Sandholm along with my guest today, Noam Brown. The Lebradas team taught their AI the rules of Poker and they gave it a reward function which was; win as much as much money as possible, and then they just told it to optimize that reward function and then they just had Lebradas train itself with simulation after simulation after simulation.

After enough training, Lebradas was ready to crush human competitors, which it did in hilarious and entertaining fashion. There's a video from Engadget on YouTube which I linked to in the show notes, and it shows the AI competing against professional humans and just destroying them. It's really entertaining.

In this episode, Noam Brown explains how they built Lebradas and what it means for poker players and what the implications are for humanity, because if we can automate poker, what can't we automate?

[SPONSOR MESSAGE]

[0:01:59.4] JM: Flip the traditional job search and let Indeed Prime work for you while you're busy with other engineering work, or coding your side project. Upload your resume and, in one click, gain immediate exposure to companies like Facebook, Uber, and Dropbox. Interested employers will reach out to you within one week with salary, position, and equity upfront. Don't let applying for jobs become a full-time job itself. With Indeed Prime, jobs come to you.

The average software developer gets five employer contacts and an average salary offer of $125,000. Indeed Prime is 100% free for candidates, no strings attached. When you're hired, Indeed Prime gives you a $2,000 bonus to say thanks for using Prime. If you use the Software Engineering Daily Podcast link, you'll get a $3,000 bonus instead.

Sign up now at indeed.com/sedaily. Thanks to Indeed Prime for being a loyal sponsor of Software Engineering Daily. It is only with the continued support of sponsors such as yourself that we're able to produce this kind of content on a regular basis.

[INTERVIEW]

[0:03:16.7] JM: Noam Brown is a researcher at Carnegie Mellon University. Noam, welcome to Software Engineering Daily.

[0:03:22.3] NB: Thank you for having me.

[0:03:23.5] JM: I want to start off by giving you and the listeners some quick context on my own background in poker, because the work that you're doing on Lebradas, the poker AI, is really close to my heart because when I was in high school I was playing poker, and then earlier on in college I was playing in some high stakes games online. This was back in 2007 when the games were really easy. Then there was some legislation that made it really hard for people to put money online. The competition got a lot harder, because the people who were fishy were not putting money online anymore and it became a competition of really good players. Around that time I started losing because I was not as disciplined. I was not as creative. My play was too algorithmic and all of the best players were more flexible. They were less emotional. Eventually, I just decided to quit and focus on school which led me to computer science which brings us to today.

Lebradas is this poker AI that you've worked on, and Lebradas is effectively more creative, more disciplined and more thorough in its exploration of the game space of poker than any human has ever been. I know this is going to be hard for professional human poker players because their time has come to past. They are no longer viable poker players compared to automation. They've been automated away before even the truck drivers.

Why is it that poker is so much easier to automate than people want to believe?

[0:04:59.4] NB: Yeah, you bring up a really good point. I think when I tell people that I'm working on AIs for imperfect-information games and applying it to poker, one of the things that I hear really frequently is, "How can AI learn to bluff like a human can?" They have this idea that bluffing is this very human behavior that an AI can't emulate.

The reality is that this bluffing behavior and really the entire strategy of poker is a very mathematical process and you can arrive at a really good strategy for poker. In fact, a perfect strategy for two player heads-up which is what we're playing. You can arrive at close to a perfect strategy using complex computer science algorithms.

Certainly, this is not easy. In fact, it's a lot harder than a game like chess or Go, and so I think people are right that this is not an easy process for an AI and that there is something far more difficult for an AI to emulate human behavior when it comes poker, but this is certainly within the realm of possibility for an AI.

I think this really highlights what we're seeing in AI in general. I think people have this idea that there are certain things that a computer is good and there are certain things that a human is good at. For a lot of these tasks, I think they categories those incorrectly. People used to think that an AI can never play chess or that it can never play Go. Until very recently, people thought that an AI cannot master poker and we're seeing that those are all actually tasks that a computer can master to greater extent than any human can.

[0:06:22.0] JM: Speaking of people framing problems incorrectly, I think this whole framing of imperfect information games versus perfect information games doesn't make a whole lot of

sense because an imperfect information game that is well-formed, it's got a probability distribution that you can map out accurately. Isn't that computationally equivalent to a perfect information game with just a very large game space?

[0:06:51.4] NB: Actually, no. When you introduce these uncertainty elements, this hidden information component, the game actually becomes much more difficult to play than a game like a perfect information game like chess or Go. For the listeners, I should clarify. What I mean by imperfect information is this component of the game where you don't know exactly what's going on or your opponent doesn't know exactly what's going on.

In a perfect information like chess or Go, both players know exactly what's going on at all times. They can see all the pieces on the board. An imperfect information game like poker, you have your cards that only you can see and your opponent has their cards which only they can see. You never know exactly what's going on. You never know exactly what state of the game you're in.

The problem there is — Go ahead.

[0:07:30.1] JM: I was going to say — You don't know exactly what they have but you can say, "Okay. There is a 1 in 52 chance that they have an ace of clubs in their hand and there's an 18% chance that they have two clubs in their hand," and you can make all of these probabilistic statements that are going to be accurate and if you have those probabilities mapped out then it might as well be perfect information game.

[0:07:30.1] NB: You're going to be correct at the very beginning of the game. There is a 1 of 52 chance that their left card is a two clubs and a 1 of 52 chance that the right card is a five of spades. As soon as they start taking actions, then those probabilities change. The way those probabilities change is entirely up to them, because they determine their own strategy. We can't make assumptions about how they're going to play.

You're a poker player. Let me give you an example. Let's say you sit down at a poker table heads-up against this player. You have 200 big blinds deep, so you have a lot of money in front you. Let's say you have $20,000 in front of you and the blinds are $1, $2. You get your hands

and the very first action of the game before you ever seen this guy play before, the first thing he does is bet all in. Now, what's the probability that he has each? You tell me. Do you have any ideas here?

[0:08:42.4] JM: Right. I see what you're saying.

[0:08:44.8] NB: You can't assume that he has pocket aces because then you're only going to call with pocket aces that he'll try to bluff you. So you can't do this reasoning of, "There is a certain probability that he has each hand, here are the probabilities." The process is far more complex than that. The correct answer of how to approach this is that you have to consider the strategy for the entire game as a whole, that's why imperfect information is so much more difficult because you cannot look at a situation or an isolation.

If I told you that, "Yeah, this guy just bet all his chips, he went all in." I ask you, "What's the right thing to do?" That really depends on how much money he could have made for each hand by not betting all in, by playing normally. When you play these imperfect information games you can't look at a sub-game in isolation like you can with chess or Go. With chess, I can just show you a chessboard halfway through the game, or I could show you there's one move to checkmate and you can play it out from there not having to know much of the strategy of chess.

In poker, it's different. In fact, in poker, as you get farther into the hand it actually becomes harder to play both for computer and for a human because there's so much more uncertainty. There are so many more hypothetical situations that might have come up and you have to consider all that we you are forming your strategy.

[0:09:55.3] JM: This is basically the meta-game that you're describing.

[0:09:58.7] NB: you could think of it as a meta-game but I see it as really the game itself. I see this as really the essence of what poker is, trying to come up with a good strategy for the entire game that your opponent can't take advantage of.

[0:10:11.3] JM: The people who worked on Go or chess, my understanding of those games, not great, but is that the game space is still too big to model completely so they have to use some

heuristics. Yes, these are, in a sense, perfect information games but since you can't traverse the entire decision tree, you have to use some form of heuristics. It seems to me that the processing of a given decision is somewhat similar to the processing of a decision even in imperfect information game where you're using some heuristics. The only difference is the heuristic in imperfect information game is you what has this player done in the past perhaps and how am I using that to traverse the information? Whereas in a game like Go, a perfect information, you can just trim off branches really quickly where you can, “Okay. This branch doesn't make any sense.”

[0:11:09.1] NB: Actually, the approach that we use for imperfect information games is I would say pretty radically different from the approach that has been used in perfect information games like chess or Go. The approaches that are used in chess and Go all function on this idea of searching through the game drive. You’re in a certain situation, you know you're in that situation, and you search for as far as you can and then eventually the game — There’s too many possibilities to consider, so you just use a heuristic to decide what's the value of this state that I've reached in my search?

Humans sort of do the same thing when they’re playing chess. We say, “You can look three moves ahead.” They look three moves ahead for all the different possibilities and once they get three moves into the future they’re like, “Well, this situation looks pretty good,” or “This situation looks pretty bad,” based on some heuristic of the game board. With poker, that approach doesn't work, and you really have to start from scratch with an entirely new approach.

Really, the essence of what we do in poker to called it a good strategy is reinforcement learning. It’s this idea of — The AI learns how to play on its own. It starts completely from scratch, playing completely at random and then it just plays itself in trillions of hands of poker. Over time it — When it finishes the hand it goes back and reviews all of the decisions that it made and it asks, “If I had done this other action instead, would have I gotten more money?”

Humans do this too? When humans are playing poker, after a hand is over they’ll ask their friends, “Well, if I had raised would you have called me?” The AI is doing exactly that on every hand. If the answer was yes, he would have gotten more money, then the AI will take that action more in the future.

Over these trillions of hands, the AI gradually improves in a way that's pretty similar to how a human improves. Eventually it arrives at what's called the Nash equilibrium. This is a perfect strategy in two player poker. Poker players might be familiar the term more as game theory optimal; GTO. It's this idea that if you are — In every game there is this perfect unbeatable strategy and expectation. If you're playing the Nash equilibrium, then an expectation your opponent can't beat you, that's what the AI is trying to approximate.

[0:13:14.1] JM: Your work is combining game theory and reinforcement learning. You teach an AI just the rules of poker, just here is how you navigate this game space, or here are the potential branches you can take in this game space. Traverse it however you like, you have to follow these rules," and then you give it a reward function and you tell it to optimize that reward function. For example, how much money you make per hand or how much money you make over a given session. Give some more color on how reinforcement learning works for Lebradas.

[0:13:50.0] NB: Yeah. Usually, that's a reward function. In poker the reward function is very simple, it's how much money you make. We give it the rules of the game. We say, "If you bet this amount, and your opponent called, and then you get to the river, the hand is over, and you have the better hands, then you win the money in the pot."

The reward function is very simple. Basically, the AI is going to have a policy. It's going to have a strategy for every situation that it could be in. If the board is 2,3,5 and it has 3,5 and the sequence, in the previous round, it had bet $200 and the opponent called, that's a situation. It's defined by the actions that have occurred, that if the bets, or the calls, and the cards that are in its hands and on the board.

For each of those decision points it is going to update the strategy every time it plays. Every time it ends up in that situation, it is going to try something and then it's going to see how much more profitable the other actions that it didn't take would have been. In those cases where it was going be more profitable by taking some other action, then it will choose that action more in the future.

One competition is that this process — Poker is a very big game, especially no limit Texas hold 'em. Were talking about 10 to 161 different decision points in the game that we're playing, which is more decision points than Adamson universe. Really, you can make arbitrary large. The reason why it's 10 to 161 different decision points is because you have $20,000 in front in the game that we're playing and you can bet any dollar amount between $120,000. You have a branching factor of almost 20,000.

You can make that even larger if you allow is, for let's say, 10 cent increments of bets instead of a dollar increments. You can make the game 10 to the 1000 something. Really, what we're dealing with here is a game of infinite size effectively and you can't come up with a good strategy for all those different situations you could be in.

What we do is we simplify the game a little bit, we say we're going to combine situations together. There really isn't that bit of a difference between betting $500 and $501 for example. Instead of coming with a policy that's unique for those two situations we're going to up with a single policy for both of them. Same thing with some of the poker hands, so there's very little — If you have ace high, it doesn't matter if you have ace high and your third kicker is a two, or ace high your third kicker is a three. Those are essentially the same hand, and for the most part you can treat them identically. You can simplify the game, come up with a policy just for those situations altogether instead of individually and arrive at a good solution faster than you would have.

There is a problem here. If you're combining these poker hands for example then sometimes the difference between a kicker of two and a kicker of three is really important when we get to the turn. The AI considers all of its poker hands uniquely for the first two betting rounds. There're four betting rounds in poker, in Texas hold 'em. For the first two rounds it's going to consider each hand individually. When it gets to the third round, before it only had an estimate for how to play for the rest of the game. When it's actually playing, it's going to recalculate its strategy and come up with a much better strategy for the situations that it could reach that point on.

This is the end game software, the sub-game software that took about up to 20 seconds to come up with its strategy during the play.

[SPONSOR MESSAGE]

[0:17:15.4] JM: For more than 30 years, DNS has been one of the fundamental protocols of the internet. Yet, despite its accepted importance, it has never quite gotten the due that it deserves. Today's dynamic applications, hybrid clouds and volatile internet, demand that you rethink the strategic value and importance of your DNS choices.

Oracle Dyn provides DNS that is as dynamic and intelligent as your applications. Dyn DNS gets your users to the right cloud service, the right CDN, or the right datacenter using intelligent response to steer traffic based on business policies as well as real time internet conditions, like the security and the performance of the network path.

Dyn maps all internet pathways every 24 seconds via more than 500 million traceroutes. This is the equivalent of seven light years of distance, or 1.7 billion times around the circumference of the earth. With over 10 years of experience supporting the likes of Netflix, Twitter, Zappos, Etsy, and Salesforce, Dyn can scale to meet the demand of the largest web applications.

Get started with a free 30-day trial for your application by going to dyn.com/sedaily. After the free trial, Dyn's developer plans start at just $7 a month for world-class DNS. Rethink DNS, go to dyn.com/sedaily to learn more and get your free trial of Dyn DNS.

[INTERVIEW CONTINUED]

[0:19:14.7] JM: Interesting. That makes a lot of sense because basically you're saying in the first two streets, the — Before you even get to the flop, and then on the flop the potential directions that a hand could go are just so multifarious. Once you get to the turn — I remember when I was playing, a lot of the best players, they would really have their — Like you said, basically, when you get to the turn, they would have all these different plans mapped out for what can happen. If I do X on the turn and then if the river comes this, here are all the different things I can do. Here are the cards I can bluff on. Here are the cards where maybe I can make a bet of up to X and get X expected value, and here are all the cards where I can make a bluff. Maybe on all of these cards I can make a really small bluff and I can actually get a whole lot of

fold equity despite it being a really small bluff. The turn and the river, there's so much more really precise planning you can do relative to pre-flop and flop.

[0:20:17.3] NB: That's correct, yeah. You can come up with a good — When the humans play, for example, when they're playing the pre-flop and the flop, they are not really thinking that hard. They just make their decision instantaneously. They have a plan already. They have planned out what they're going to do in every situation, and the AI is doing the same thing. It's already decided what it's going to do in all the situations. That might come up on the pre-flop and the flop the first two rounds.

For the turn of the river, it has to think more carefully and analyze the exact situation that it's in. For the pre-flop and the flop, it's only really coming up with an estimate, a reasonable — A sort of rough sketch of how things might play out on the turn of the river. It turns out that in our experiments that's actually all you need. You really only need that good estimate. Certainly, your performance is going to depend on how good your estimate is, but we find that it gets very good performance and the estimates it comes off with are very accurate.

[0:21:02.7] JM: That's so funny because back when I played, there were all these people who would be successful with different styles. The "styles" that they would were often different in terms of what is their pre-flop range, how much "floating" do they do on the flop. A float is where you take a turn where you just have a backdoor flush draw or you've got bottom pair, something really weak where you're just like, "Yeah, let's see what happens."

The different styles were often articulate by that, but all of the players who were successful had extremely sophisticated turn and river play. I think that falls in line with what you're saying about the general truths of how you can build a poker AI.

[0:21:43.8] NB: Yeah, definitely. I think that the strength of the AI is really its ability to play the turn and the river extremely well, far better than a human can. There's some really advanced things that you can do on the turn and the river that most humans — In all streets, that the computer can do way better than humans.

For example, you might be familiar with blockers. It's this idea that if there's three spades in the board and you have a spade, then your opponent can't have — You have the ace of spades, let's say. Then your opponent can't have the ace high flush because you block that hand.

This plays a big part in a lot of bluffing, because you want to bluff in situations where you have the ace of spades in that situation, because you can represent the ace high flush knowing that your opponent can't have it. The AI is really good at this, far better than any human, and it was one of the main strengths of the AI in the competition.

[0:21:43.8] JM: Really? Even in just no limit hold 'em?

[0:22:35.4] NB: Yeah. In no limit Texas hold 'em, blockers — At the top level of play, blockers are very relevant and, really, the difference between beating that best the world and beating recreational players.

[0:22:47.4] JM: Okay. Because I knew blockers were like crucially important for PLO, the Omaha —

[0:22:53.3] NB: Yeah, for pot-limit Omaha — Certainly, for pot-limit Omaha, it's more relevant. That's true. For Texas hold 'em, when you talk to these top-level pros and you ask them, "Why are you making this decision?" They're always thinking about blockers. That's what tilted from being a bluff hand to being a fold hand.

[0:23:09.1] JM: Right. I guess a lot of these experiments that you were doing were deep stacked also, right? I guess in deep stack, it matters a lot more, the blockers.

[0:23:18.0] NB: Yeah, definitely. We're playing too much of big blinds deep.

[0:23:20.3] JM: Right. Okay.

[0:23:21.8] NB: which is great, because it's more challenging for the AI in a lot respect, because it makes the game much bigger, so many more possibilities that could come. If we were playing, for example 50 big blinds deep, then it's actually much easier to make AI that's close to perfect.

With 200 big blinds deep, it was a real challenge, but also allows for a really high level, really advanced strategic play because when you're in situations where there's so much money in the pot, there is far fewer hands that a person could have. If there's a raise and a re-raising, and then another re-raise, then they're only going to do that with either really really good hands or a hand that could really represents very effectively being a really really good hand. This blocker analysis becomes very relevant in those situations.

[0:24:06.3] JM: We've been doing some shows around deep learning recently and I'm starting to get a better understanding for this space. I think maybe a couple of weeks ago I might've asked, "Why aren't you using deep learning?" I think more recently, I'm starting to understand deep learning is really useful for building these layers of abstraction for collections of data that are really hard to understand. They're really deep and hard — Not that term deep that I just used that have anything to do with deep learning, but like a complex image is really hard to even understand how do you parse this image, and so you buld up these different layers of understanding the image. First, you understand the pixel level and then you understand at the edge level and then you understand it at the — I don't know. Some small unit level.

In poker, for example, the game is completely well-defined. It's just quite a different set of problems, so deep learning doesn't really apply to this sort of problems. Is that correct?

[0:25:06.0] NB: We did not use deep learning in our AI. The focus of our research has been on reinforcement learning and trying to extend the results of reinforcement learning to improve information games. This has been a real challenge in AI going back for a long time. People knew how to make AIs for perfect information games, like chess, and it sort of had an idea for how to do for Go also. In 2016, they really figured it out.

With poker, it had always remained this challenge because it's a fundamentally different kind of problem. That's really what we address with our AI. We don't use deep learning with it. That said, the methods that we're using are not incompatible with deep learning. You could throw in deep leaning, and depending on how you implement it. Yeah, it could help with some aspects of the AI.

I think that it's not like there's a lot of different aspects to AI. Deep learning is one area, but there's a lot of others as well and are not all incompatible. I think that deep learning has gotten a lot of attention in recent years and for a good reason, but it's certainly not the only area of AI.

I think this competition, and Lebradas really highlights that. You can see AIs advancing in a lot of different areas, and certainly reinforcement learning, which is what we're using, is a really quickly advancing field.

[0:26:16.9] NB: One thing I could imagine is if you had two AIs the were competing to recognize images better, then for each AI, you've got all these different variables that you can tune to make the best image recognition algorithm and then you can use reinforcement learning to train the weights on those different variables. How would you apply deep learning and reinforcement learning together to a poker AI?

[0:26:47.8] NB: One area where deep learning might be helpful — Again, we did not need deep learning to beat the top humans, but for imperfect information games in general, maybe other games, it could potentially help.

I mentioned earlier that we combined some situations together. For example, if you have a bet of $500 and a bet of $501, we treat pretty much the same. We do that when we're coming up with these estimates for how to play later on.

We have techniques for how to do that, but alternatively you could use something like deep learning to figure out how to combine those different situations together and simplify the game. That might be an area where deep learning could be applied in the future.

[0:27:29.7] JM: Can you talk any more about what the implementation of that would look like?

[0:27:34.1] NB: I can't really comment too much because I haven't tried. This is just speculation about what directions people might go in the future.

[0:27:41.6] JM: I was watching these videos of Lebradas playing people and it was funny to see these strategic manifestations that occurred because it some of the stuff that Lebradas does are

things that people would explore at high-stakes poker but they wouldn't explore them as thoroughly. Stuff where really tiny bets or huge over bets. These things that people like Prahlad Friedman did, Stu Unger did this kind of stuff. People like — Who was that really crazy high-stakes player, the guy that was dominating for a while? Anyway, I forgot his name. There's a lot of people like that.

I felt like a lot of times they didn't really talk about the mathematical justification for it. Maybe they did have a mathematical justification and they just didn't even — Maybe they didn't understand it or maybe they just didn't tell anybody because it was their secret sauce. I think, today, probably the math of over- betting or under-betting is — Maybe it's a little better understood. Can you talk about why Lebradas makes these weird bets? Why they're actually mathematically justified?

[0:28:48.3] NB: Yeah, this is one of the really cool things about the AI is that since it hadn't learned how to play from human data, it learned by playing against itself. It started from scratch. It was able to come up with these really innovative strategies that humans don't really employ, and the pros that we played against had a lot of trouble responding.

As to why it shows to make these huge over-bets, where it was betting eight times the pot in some situations. I don't know what's its thought process was. I'm not a pro poker player. I didn't tell it to do those things. I just gave it the rules of the game and after playing trillions of hands it decided, "Hey, that seems to work out really well."

[0:29:24.7] JM: Did it do that on the turn or on the flop?

[0:29:27.5] NB: Yeah, it would do it on the turn. It would sometimes do it on the flop. It was great. The pros that we're playing against said, "This is something that they really have to start using in the future," because it really allows you to grow the pot very effectively when you have a strong hand. Of course, you need to balance that, because you can't just do these huge over-bets when you have good hand, because then your opponent would always fold. You have to mix in bluffs as well.

I think that that's a real challenge for humans. I think they're really afraid to make these huge over-bets when they have a bluff. Understandably, right? Humans don't want to — Is the subversion to being to betting $20,000 into a $200 pot when you have three high, right? The AI obviously has no fear.

[0:30:07.1] JM: What's funny is because if you start to do — It's almost like humans had this silent agreement just to not over bet before the river, because if you do that, the effect is that it increases the risk of ruin for everybody, perhaps. Maybe that's an over overambitious statement there, but it seems like if you do — That would happen, right? You probably increase risk of ruin for everybody and nobody wants that.

[0:30:32.4] NB: Yeah. Honestly, the person we played against said the same thing. They said that when they're playing against other humans, there are sort of these unwritten rules that people just don't do certain things, and nobody really thought that it was that much of improvement to actually do those things. It just that — They thought it just increased the variants, it increased the chance of losing tons of money, so they just never did it, and the AI disrespected all of those rules, and they really hated the AI for that.

[0:30:59.2] JM: It was funny watching these pros. By the end of it, they were just so exhausted and so beaten down. They started doing this strategy. They're like, "Let's just three bet it 80% of the time.

[0:31:14.5] NB: To be clear, they thought that was a good strategy. They analyzed the hands on the previous days and they determined that — Based on the data, they thought three betting 80% of the time was going to win them a lot of money, and it turns out they were very wrong. They had a reason for it.

[0:31:29.2] JM: Did that just increased the velocity at which they lost money?

[0:31:32.6] NB: Basically, yeah.

[0:31:33.3] JM: It's so funny that these two players coordinated still could not beat it. Although I guess that that make sense because Lebradas is effectively an infinite number — Well, it's

some number of robots that are collaborating with each other just like comparing advice and stuff.

[0:31:47.9] NB: Yeah. They were all actually playing against the same copy of the AI. It was interesting. The AI was improving based on play against the humans. It was looking at what situations were the humans putting it in. Overnight, it would learn how to better respond to the situations. It wasn't adjusting — When the media first heard about this, they immediately latched on to this is as the AI adapting to the opponents and learning from the opponents and trying to figure out how to beat them.

That's actually not the case. The AI was looking at what situations the humans are putting it in and improving those situations to try to get closer to this perfect strategy, this unbeatable strategy.

The humans, they picked up on this and they then decided to coordinate with each other, so that instead of using the same sizes and putting the AI in the same situations among all four of them, they would use different sizes. They would use different bet sizes so that the AI would have a harder time learning how to respond to all of them.

I got to hand it to the humans. They put up a huge fight. They put up an incredible fight and they really gave it their all. They studied every night trying to figure out as a team how to take down this AI.

[0:32:50.6] JM: They were playing the bot and then the Lebradas would — There's some off-line batch learning that it would go through every night. Can you talk more about that? The cycle of learning.

[0:33:01.9] NB: Yeah. The idea here is that on the first and second round, the AI considers a bunch of different sizes, a bunch of different bet sizes both for itself and for the opponent. Like I said, there are 20,000 different possibilities, because you can bet any dollar amount between 100 and $20,000, but it doesn't consider all of them. It will round, for example, a bet of $501 to a bet of $500.

That's obviously not a perfect strategy, $501. A bet of 501 is different from a bet of 500, and the optimal strategy for those two situations is certainly different. The AI would see what bet sizes were the humans actually using on the first and second round and if they were frequently using similar sizes, like 525, 526, 527, something like that and those sizes were far away from a size that it already had in its abstraction that it was already considering. Basically, if this rounding error was pretty large for the situations that the humans were putting it in, then it would learn overnight how to better respond to that situation in particular rather than the rounding it to the nearest size.

[0:34:06.8] JM: Can you explain how much of that was hardcoded and how much of that was strategic algorithms that the AI learned?

[0:34:06.8] NB: This is all the strategic algorithms that the AI learned. Very little of this is handcrafted. When it was deciding what actions to improve upon, for example, it used an algorithm to decide, “These actions that the humans are using are far away from a size that I'm already considering and they seem to be betting this amount a lot, so I should really focus on this situation in particular to improve?”

Some things are handcrafted, and I think that this is an area for improvement. In particular, the sizes that the AI considers, it considers a bet — It might bet a quarter pot, or it might bet a half pot, or it might bet one times the pot. Those sizes were, for the most part, not entirely, but for the most part were determined through a handcrafted choice. A human, basically myself and adviser, decided on those sizes that it would consider. Then among those sizes it would choose which ones to use. That way it didn't have to consider all 20,000 different sizes, t would only consider maybe 10 different sizes to choose from.

We do actually have an algorithm that can determine those sizes automatically, and I think in the future it will be really nice to use that instead, and we did use it to, for example, determine the first and the second bet sizes. For example, the algorithm determined that if the optimal opening bet size in heads-up no limit Texas hold 'em is 0.75 times the pot, 2.5-X, which coincidentally is something that humans are now using pretty frequently. We used that size in the abstraction. Some other ones were handcrafted, yeah.

[0:35:43.7] JM: What programming languages and machine learning frameworks are using?

[0:35:48.9] NB: The algorithms were written in C++. We used MPI to communicate between the nodes during the equilibrium finding process, basically the component where the AI is playing against itself over trillion of hands. We didn't do that on a single node. We actually distributed it over about 200 nodes on supercomputer. Then the actual runtime agent that handles the communication between the human and the graphical user interface, all these stuff, that's written in Java.

[0:36:13.1] JM: I don't know anything about MPI. I feel like I've heard about that before. What is MPI?

[0:36:18.6] NB: MPI is a protocol. It's a framework for communicating, basically it's an API for communicating between nodes in a distributed process. You could think of it as a much faster version of TCP/IP years, like socket communication for example when they're running on a local cluster. It's really low level, that's the key difference between this and a lot of other communication protocols, is extremely low level. You're saying I want to send a message of the size and the bytes are this, basically.

It's really fast. It's not robust to errors at all. You have to handle on your own. If you're doing something that's — Each hand that the bot is playing takes milliseconds, a few milliseconds at. We have to have something that's extremely fast, and MPI is the right choice for it.

[SPONSOR MESSAGE]

[0:37:11.4] JM: Artificial intelligence is dramatically evolving the way that our world works, and to make AI easier and faster, we need new kinds of hardware and software, which is why Intel acquired Nervana Systems and its platform for deep learning.

Intel Nervana is hiring engineers to help develop a full stack for AI from chip design to software frameworks. Go to softwareengineeringdaily.com/intel to apply for an opening on the team. To learn more about the company, check out the interviews that I've conducted with its engineers.

Those are also available at softwareengineeringdaily.com/intel. Come build the future with Intel Nervana. Go to softwareengineeringdaily.com/intel to apply now.

[INTERVIEW CONTINUED]

[0:38:04.3] JM: Help me understand how much of this is offline and how much of it is online? When you have all the stuff that runs at night and is retraining the model that you have just this model that sits in memory somewhere, or what kind of learning you have to do even when Lebradas is competing?

[0:38:25.7] NB: If you look at the number of core hours that we used, the amount of computational time spent, the majority of it, it's about, I would say 60% of it was spent online during the competition. Yeah, about 40% of it was spent before the competition doing this reinforcement learning against itself kind of think where it started completely scratch and it played trillions of hands against itself and it learned a good strategy. Of course, it came up with a really really good strategy only for the first two rounds and estimated a strategy for the third and the fourth round. The 40% spent during the competition was mostly spent on coming up with the good strategy for those 3rd and 4th rounds in real-time.

The amount of power that we're talking about is somewhere in the ballpark of 20 million core hours. This is not a cheap operation. We're running on — To train the AI, we used 200 nodes. Each node having 28 course. That's a lot of power.

During the actual competition we had a copies of the AI running, each one used 50 nodes and each node had 28 course.

That said, this technology is going to improve very quickly. This is where it is as of three months ago, but this was developed just by myself and my adviser. If I had more time to code up the AI, we could get this running much faster. I've told people this before, that I really think that within five years you could see this kind of technology deployed on a smartphone and doing just as well beating all the humans of the world at a game like heads-up no limit Texas hold 'em.

[0:40:00.6] JM: Speaking of deployment, how much of the players online are bots these days?

[0:40:05.4] NB: I don't know. I don't play online. I have never run the bot online or any other bot online. From what I've heard, the poker sites are very effective at cracking down on bots. They have a lot of security measures in place. If they see, for example, that you've been playing for 24 hours straight, then they'll throw up a captcha that ask you, "Are you a human?" They'll have sophisticated techniques, the trip up computer vision algorithms that might be trying to play automatically, or they'll look at what programs are running on your computer.

In fact I've even heard that if they really suspect you're a bot, let's say you've won a really big hand, for example, they will call you up and ask you, "What was your thought process during that hand?" The reasoning being that if you're actually a human and you were in a huge pot, you're going to remember how you played that hand. If you just have the bot deciding for you, then you're not going to remember the details necessarily.

Certainly, there's a huge risk to running an AI online. If they suspect that you're a bot, they're going to confiscate your money. That's the best case scenario, is that they'll confiscate your money. The worst-case scenario is that you're doing this in a state where it's illegal and you actually go to prison.

I mean I think there are bots online certainly. I have no idea how prevalent they are. I think it's going to become more of a problem in the future as this technology becomes more widespread. We're already seeing that there are there — There's a lot of tools for poker players in general, right? You have pre-flop tables, you have end game solvers, all these things, and I think that once the research for Lebradas gets out there and become more common in the poker community and better understood by the poker community, then these tools that humans are using are going to improve. Even if you're playing against a human, they're going to be assisted by an AI. It's substantially going to be like you're playing against an AI.

[0:41:44.8] JM: I think I would be more afraid to play on the 50 cent dollar no limit tables today than the 510, because 50 cent dollar is probably like people who have bots. They're running 800 bots distributed along the 50 cent dollar tables, and then poker stars is really focused on like the 510 tables for their anti-bot measures. I don't know. Who knows?

[0:42:07.9] NB: That's probably true. Certainly, the AIs that are — The AIs before Lebradas. Even two years ago, they were able to beat at least 95, even 99% of online poker players.

[0:42:20.0] JM: Yeah, it's just not that hard of a problem.

[0:42:22.6] NB: It's not that hard to beat bat players. That's easy to do with an AI. You could even take a human that's like a pro poker player and he could hand code some rules for how to play, and that would beat most of players at the really low stakes, because there's just a lot of people out there that are willing to throw their money away for entertainments.

At the high-stakes, then that's a different story. Certainly, before Lebradas, before three months ago, no way I was able to beat top players, or people that that really took it seriously.

[0:42:49.4] JM: The reason I led off this conversation with a personal anecdote, I don't tend to do that in each show, but the reason I do that is because poker — I had to leave the game because I was no longer a viable career to me. For a while, that was really difficult because I was like, "What am I going to do now? I focused my entire life on poker," and I'm sure through the Lebradas project, you've met plenty of other poker players who that's all they do. It's like their life. It is their culture. It's everything they do and they're going to have to abandon that, because it's no longer going to be a viable career. It's been dying as a viable career for many years since the boom.

The long-term implications for me were quite positive, because I had to reinvent myself and that was great. I'm wondering what your take is on that, because all of these people are going to be automated out of things like truck driving in the near future, and it seems like this is actually an opportunity for tremendous reinvention along these people. I also know a lot of poker players who have never reinvented themselves, they've just kind of languished in gambling and now they're addicted and it hasn't turned out well. I don't know. I think you're in an interesting position to comment on the pros and cons of white-collar jobs being automated away.

[0:44:03.3] NB: Right. I think you bring up two points. One is automation of poker players, basically, and automation of other professions in general. With regards to poker, I think you're

absolutely right, that the poker broom has ended. It's never going to die. Poker is never going to die, but certainly we saw this effect where you had — It's like in any ecosystem where the smallest level of the food chain dies out. It's working its way up the chain and the slightly larger fish are dying out of that. After that, the bigger fish die out. Eventually, the sharks are going to go hungry, and that's sort of what we're witnessing now and it's certainly — I admit that developing AIs like these is not helping in that situation. I think that it's going to hurt online poker, in particular, pretty hard.

I still think that AIs are not going to be as widespread. This AI, for example, it plays heads-up. The techniques could be extended. The reason why we focus on heads-up is because if you really want to figure out who's better, you really have to play heads-up, because then you don't have any issues of collusion or interactions between other players. Lots of the techniques could be extended to more than two players and it will probably do just as well. I think it will be another year at least until this kind of technology really spreads out into six-max and nine-max games.

Eventually, online poker is going to face serious problems. I still think, because live poker at the casino is going to do well. In fact, I think one of the reasons why we did this whole competition at the Rivers Casino in Pittsburgh. They were phenomenal, by the way. I think one of the reasons why they were willing to host this kind of events is because it really highlights that, "Hey, you might not want to play on online poker tables anymore. You might want to come here instead where you know that you're not playing against an AI."

For the people that are leaving poker, I really think that it's is not a bad thing. Yeah, poker is fun, but I think it's something that you don't want to necessarily do for your whole life unless you're at the absolutely top of your game. It's very difficult to make a living and to make a whole life out of playing poker. This could be a very good opportunity for professional poker players who are not at the very top of their game to reinvent themselves, like you have done.

With regards to automation general; automation of truck drivers and all these jobs — Yeah, AI is sadly going to replace a lot of human jobs. I say sadly. It's sad in one sense that a lot of people are going to lose their professions. On the other hand, it's going to create a lot of opportunities is well.

I say it's very similar to the Industrial Revolution. The Industrial Revolution removed a lot of jobs, but at same time it created a lot of opportunities and a lot of wealth for society. I think it's important that we recognize that there is going to be winners and losers and there's definitely going to be more winners than there are losers, but it's important to compensate the losers and make sure that they're not getting an unfair share of dis-advancement, that they're compensated for their loss by the winners in this audition process.

[0:47:03.0] JM: I want to wrap up with one more question. There's the Andrew Ng camp of thinking that general artificial intelligence is like overpopulation on Mars even though it's a tale risk. We shouldn't even worry about it right now. Then there's the camp of Elon, Bill Gates, Nick Bostrom that artificial general intelligence threatens to turn us into paperclips. Are you afraid of artificial intelligence in any degree or any framing?

[0:47:37.2] NB: It's an interesting question. This is something that's really — It's sort of been delegated to science fiction for a long time and I think that we're seeing such huge advancements in AI that maybe the time when AI surpasses human intelligence is not as far off as we thought.

That said, I still don't think that AIs are going to surpass human intelligence anytime soon. I think that this is not going to be a real practical concern for 50, or 100 years at least. Yes, we're seeing AIs beat humans at chess and Go and poker now, but you're still not seeing AI writing a prize-winning novel, you're still not seeing an AI coming up with really cool and elegant mathematical theorems.

I think what we're seeing is that some things that we thought only humans were good at. We're actually seeing now that computers can handle those tasks as well, but there are still many more tasks that humans continue to excel at and that AIs are really really bad at. I think that we're just not really focusing on those because they're not as exciting.

I'm not terribly concerned about in the near future. I think that we should be focused on how these advancements in AI are going to enrich our lives. Maybe one day we're going to have to worry about what intelligence really is, what is it means to be an AI versus human, but I think that's going to be a question for our grandchildren, not for us.

[0:48:51.5] JM: Okay, Noam. I want to thank you for coming on Software Engineering Daily. It's been a fascinating conversation. I'm excited about Lebradas.

[0:48:58.5] NB: Yeah, thank you for having me.

[END OF INTERVIEW]

[0:49:07.5] JM: Indeed Prime is a sponsor of software engineering daily and they worked with us to create some tips for the hiring process. Today's tips are about hiring management. If you're a hiring manager, I've got seven tips that I think you might want to consider. These tips are sponsored by indeed Prime, but I wrote them myself.

Tip number one; optimize for growth. If you are hiring people, you don't necessarily want to hire the person that is right now ready for the role that you're hiring for. That would be great but oftentimes you can't find that or hiring that person could be super expensive. Maybe take a risk on somebody. Take a risk on somebody who just graduated who seem super hungry and optimize for growth in the candidates that you look for.

Tip number two; hire for strength, not lack of weakness. Oftentimes people have super exceptional strengths but they've got a collection of weaknesses, like interpersonal skills, or maybe they're not great at writing unit tests. People can learn these things, or you can get those things from somebody else who has a strength in them. It's much harder to cultivate an extreme strength of a candidate.

Tip number three is left a candidate propose a hiring procedure. Oftentimes the hiring procedures that are boilerplate that you could find online are not modern and they don't let a candidate optimize for the things that they're great at, why not give the candidates the opportunity to structure their hiring process? If you say to them, "Hey, work with me to develop a hiring process just for you." If they can't do that, maybe you shouldn't even be hiring them in the first place.

They should have enough creativity to say, “Okay, let me build you a side project that displays quality X,” or “Okay. Let's do this structure of programming problems, “and then they’ll give you some ideas for programming problems they might want to work on and then you could work on them together. I think these kinds of things can really show the best features of a candidate.

Tip number four is give the candidate an opportunity to show strength. This really goes with tips number two and three. You really want to give the candidate the opportunity to show their best side, and oftentimes these super limiting hiring processes, like white-boarding, don't really let a candidate get creative and show their best side and it makes people nervous when they don't get to live in their own skin, when they have to live in this manufactured world of whiteboard problems.

Tip number five is if you want servility, you can focus on servility and boilerplate programming problems. Sometimes it is good to have these white-boarding problems because you have extremely well formed problems in your engineering staff. Maybe you've got this very specific set of re-factoring things that you need done. If that's the case, if that's what you’re hiring for, then maybe you should do white-boarding problems. Maybe you should have somebody find all the subsets of some collection of numbers that sum 215.

I would say that's it's generally better to hire creative engineers, and that's tip number six. If you want creative engineers, focus on flexibility and creativity. That's why I'm suggesting that maybe you should even let the candidate propose their own hiring process because if you present that to them then an honest candidate will say, “Okay. I’m going to invent a hiring process on the fly that's going to test my creativity. It's going to present to you that I'm creative and it's can be fun for both of us.”

If that's too complex of a task for this engineer, then maybe they're not a great fit for a creative engineering role. A lot of the hardest engineering roles take a great deal of creativity, they take great deal honesty, and they take a great deal of execution and communication skills, and I think the idea of flexibility and creativity in the hiring process mirroring the way that the job will proceed which will also be creative and challenging. It's a good way to set yourself up for a good relationship with that engineer if everything goes to plan.

The final the final rule, rule number seven, or tip number seven, if you are hiring for boring legacy programming work, be honest about that. I know a lot of engineers, myself included, who have been kind of tricked into doing boring work. You get hired, you show up on day one and you've gotten sold this really interesting project, “Oh! You're going to build an antifraud system for analyzing all the transactions on our marketplace.” You show up and it's, “Actually, you’re re-factoring this terrible legacy thing.”

This is actually a really common practice in engineering and it's totally counterproductive. There are people who want to do that kind of legacy re-factoring work. In fact, we did an entire show about it with Andrea Goulet. Some people like doing legacy programming and that's okay, and companies that hire for that kind of work should be honest about that.

This concludes my seven tips for hiring managers, sponsored by Indeed Prime. If you want to support Software Engineering Daily and check out a new way for hiring, whether you’re a hiring manager or an engineer, go to indeed.com/sedaily. Thanks again, Indeed.

[END]