Episode 04: Shall We Play a Game? Rise of the Machines

Show Notes

For about as long as there have been computers, there have been computer programs that play games. This episode considers some of the history of game playing computers, and how that has shed light on the nature of human intelligence.

Game References

Chess, Go, Jeopardy!, Pong, Tic-tac-toe, Uncharted, Video Olympics

Research References

Isaacson, W. (2014). *The Innovators: How a Group of Inventors, Hackers, Geniuses and Geeks Created the Digital Revolution*. Simon and Schuster: New York: NY.

Licklider, J. C. (1960). Man-computer symbiosis. *IRE Transactions on Human Factors in Electronics*, *1*, 4-11.

Metz, C. (2016). What the AI behind AlphaGo can teach us about being human, *Wired*.

Montfort, N., & Bogost, I. (2009). *Racing the Beam: The Atari Video Computer System*. MIT Press: Cambridge, MA.

Transcript

Hello! This is episode four of Cognitive Gamer. I am your host, Steve Blessing. In this episode we will get a little philosophical as we talk about computer programs that play games, and how that has informed our knowledge about the nature of intelligence and what it means to be human. I teach an upper-level undergraduate course that I call Thinking, which deals with higher-level cognitive issues like reasoning and decision making. Being an upper level course, we'll have class days devoted to discussion, and often we'll have readings that ultimately come back to these issues. What does it mean to be smart? What is intelligence? Will computers ever be as smart as humans, whatever that means? They are good discussions, and it has been interesting to see how student's views have changed over the dozen or so years that I have taught the course. As a group, my college students are getting more receptive to the notion that computers will one day exhibit a true intelligence. But, I'm getting ahead of myself. Let's talk a little about the history of these investigations.

For about as long as there have been computers, there have been computer games. As soon as the bits settled from breaking codes and calculating ballistic equations during World War II, computers were put to task playing tic-tac-toe, spacewar, and of course chess. Just like now, games have always been used to task the capabilities of computers, pushing the boundaries of what is possible. As soon as a new technology is devised, like 4K and HDR television, game creators fill the space with the latest and greatest consoles and games, reaching for greater and greater fidelity. I'm sure some people chide this relationship between games and technology, believing that games are too frivolous to deserve the latest and greatest tech coming down the pipeline. However, Demis Hassabis, one of the creators of Google's AlphaGo program, which we will talk about later, noted that, and I quote, "Games are kind of a microcosm of the outside world; that's why games were invented, that's why humans find them fun to play." End quote. That's a wonderful notion for a number of reasons, but it gets at the heart of why games and computers are meant for each other. By getting a computer to play a game, we'll understand a bit better about the nature of the world, and about the nature of being human.

A lot of the effort that has been spent programming computers to play games have concentrated on the classic games, like chess and backgammon. These are referred to as abstract games, as they don't have a strong theme associated with them, like building trains, owning property, or eradicating diseases. Maybe from the start that makes them easier to program in a computer, but even games with a strong theme can be boiled down to its essence, like resource management or set collection. But regardless, we'll start our discussion with attempts to create computer programs that play these abstract games.

Let's start with the simplest of abstract games, tic-tac-toe. Once you get past age 5 or 6, you probably haven't played much tic-tac-toe. It's a very simple game, and you quickly work out how to play every game to a draw. Part of why it's so easy is because there are, relatively speaking, so few possible games that can be played, less than 30,000 when you consider symmetry. Even very early computers, with their lack of memory and processor power, could be programmed with the knowledge needed to play every game to a draw. Not surprisingly, given that no one in the double-digits of age actually plays tic-tac-toe willingly, that turns out not to be very much fun. The programmer of one early tic-tac-toe program, Relay Moe, did something interesting. He gave it the capability to make mistakes, to make a non-optimum move. That made the game a bit more fun, and to note for our current conversation, a bit more human like.

Starting pretty early on, chess was the game that most programmers would test their mettle against. From the late 1940s till the late 1990s, that was the main game that people interested in artificial intelligence would program a computer would play. Unlike tic-tac-toe, chess is very complicated. Claude Shannon gave a lower-bound to the number of possible legal chess positions to be 10 to the 120 power. Indeed, Alan Turing wrote a chess program that was not able to be realized in the limited computers available in the late 1940s. That makes it essentially impossible, given the computers at the time at least, to compute out all possible moves. So, to get a computer to play a decent game of chess, the program would have to be quote-unquote smart in order to weed out the possibilities. The race was on to see if a computer program could beat the best human chess players. Some people claimed that it would never be possible, because computers are just dumb machines, and would never be able to play chess to the level that humans can play at. Others claimed that it would be done in 10-15 years and a new age of machine intelligence would be upon us. Others claimed that once a computer beat the top human grandmaster, however long it took to design the program, that it would be definitive proof that a machine intelligence is possible that would match, if not surpass, that of human intelligence.

All of these people were wrong. A computer program has beat the top human chess grandmaster. It took about 50 years for that to happen, though, with IBM's Deep Blue winning against Gary Kasparov in 1997. But, in the end, while it shed a little light on human intelligence, it definitely didn't show that humans were going to be outsmarted in a general sense anytime soon. Deep Blue was very good at playing chess, obviously, but nothing else. And, for what it's worth, it doesn't play chess like a human. Human players, even grandmasters, don't look ahead very many moves, at most 2 or 3. That's all our brains can handle. We work much more on pattern matching, likening current board positions to past board positions, and making moves that are similar to past successful moves. Deep Blue did that too, of course, but relied much more than humans do on looking ahead several moves. That is what gave it the edge on Kasparov.

A lot of hay is made in science fiction about neural networks. Just about every smart robot is said to have a neural network for a brain. That's what Lieutenant Commander Data had in Star Trek: The Next Generation. But then again, the writers also said he worked by heuristic algorithms, which is a contradiction in terms, so I don't know if I would believe them. However, neural networks are a real way of computing, and they have promise of mimicking human intelligence. Neural networks, which also goes by the names parallel distributed processing and connectionism, relate back to a way of modeling how human neurons, the brain's building blocks, process information. The early neural networks did so very simply and abstractly, but there was a connection between how these artificial systems worked and how actual brain circuits worked. However, like the early promises made by the pioneers of symbolic artificial intelligence systems, the early promises of the connectionists also didn't pan out. These networks weren't a panacea that will solve any problem that they were trained on. But, as stated, they are similar to how humans process information, and as we have gotten more sophisticated in understanding how our brain circuits are actually wired, we have developed more biologically plausible neural networks.

Neural networks are behind a lot of today's technology. Whenever we talk to Siri or Alexa or any other voice recognition program, that signal is probably processed by a neural network in order to figure out what we said. At an important level, neural networks are good at the same thing human brains are good at, finding patterns in information. When someone says "Hello" to us, no matter if it's a high-pitched voice or a low-pitched voice, or an American accent or an English accent, we can recognize the word because that acoustic pattern that we hear is matched to the closest pattern we have stored in our brain. That is how neural networks work, by taking in information from the outside world, and finding the patterns within that signal. That can make them particularly good at learning how to play games, because in order to get good at most games, that's what you need to do, figure out the typical patterns in the game and figure out what to do when a particular pattern presents itself.

This is how AlphaGo figured out how to play Go. AlphaGo is a neural network that has been exposed to and has played countless games of Go. Over all that experience, it has learned what a good move is by figuring out what moves lead to success given a particular configuration of pieces. In March 2016, AlphaGo played a 5 game match against the top Go player in South Korea, Lee Sedol The matches were covered in great detail by Cade Metz in Wired magazine. Not growing up in the culture, I don't have a great understanding of the aura surrounding the game, but from hearing accounts, there's a reverence for the game that surpasses what we had for chess. The grandmasters of Go are considered not only intelligent, but the thinking is that to be good at Go there's as much of an art to it as a science. Part of that is due to Go's complexity; remember, chess' game tree is about 1 with 120 zeroes behind it; Go's game tree is a 1 with 360 zeroes behind it. The belief was that only humans could play go at the highest levels, given that complexity. In the end, though, AlphaGo bested Lee Sedol four games to one in that March 2016 match.

There have been computer programs that have tackled playing non-abstract games as well. Perhaps the most famous of these is Watson, which did a great job playing the quiz game Jeopardy against human opponents. Watson was another project from IBM, and in 2011 played 3 games against the top human opponents. Watson had little trouble winning all the games. Jeopardy is interesting, because in order to play it you have to understand language, and provide human language responses. Furthermore, understanding Jeopardy clues involves complex language processing, because often the clues involve mashing together two or more concepts in interesting ways. Understanding language to the degree that we do is a hallmark of human intelligence, because no other animal produces the type and level of language that we do. To have Watson play Jeopardy in real time and do as well as it did was quite an achievement. Watson used a variety of techniques to play Jeopardy, and those techniques have been deployed in other tasks, such as analyzing medical texts in order to provide sophisticated diagnoses given a list of symptoms.

People often talk about how good the artificial intelligence is in computer games. I recently played through the Uncharted series on the Playstation, and one can see as you go from the first game to the fourth game in the series how much better your computer opponents got as you went through the games. There's a somewhat fine line that programmers need to walk here as they make the AI better in these sorts of games. It can't be too easy, or it would be no fun, but it can't be too hard either, because again, it would be no fun. Programmers have thought about these issues from the earliest of days. I recently read a book called Racing the Beam by Nick Montfort and Ian Bogost about the early Atari VCS system back in the late 1970s and early 80s, the first really successful home video console. An early game for it was Video Olympics, a collection of Pong type games. The programmer, Joe Decuir, created some one-player variants, where a human played against a paddle controlled by the computer. The computer could of course be programmed to play a perfect game of pong; all it had to do was match the vertical position of the ball, which it could do without error. Humans can't do this of course, which is why it's fun for two humans to play against each other. In order to make it fun to play against the computer, error had to be programmed into how the paddle tracked the ball. But, not just any error, because again, if it looked like the computer was playing randomly, it would be no fun. At its heart, this is what the enemies in Nathan Drake must also do, provide enough of a challenge to be interesting and look like they are being controlled by a human intelligence. If they are deadshots and always hit you, that would be no fun, but if they look like they run around randomly or only follow their pre-determined paths, that is not interesting either.

Where does this lead us in figuring out human intelligence? To me, the issue comes back to one of understanding. Does the computer program ***understand*** what it's doing when it's playing your opponent in Video Olympics or the bad guys in Uncharted? No, it's just following its program, calculating in the error in movement and aiming that the programmer told it to do in order to look more human like. What about Deep Blue or even AlphaGo, do those programs *understand* the games they are playing? Probably not, and definitely not in the same way that humans that play those games understand them. This is a classic critique of artificial intelligence, one that comes to the fore in a thought problem referred to as the Chinese Room Argument. This was first posed by a philosopher by the name of John Searle. To picture the problem, imagine you are sitting in a simple room, with only 4 walls, no windows or door. There is a slot on one wall, and a book and pen on a table. A slip of paper comes through the slot. On the slip is Chinese writing, which you do not know. Flipping through the book, you notice the symbols on the slip are in the book, with other symbols beside them. You write those other symbols on the back of the slip, and put it through the slot. A short time later, another slip of paper comes through the slot, and you find those symbols in the book and write down the corresponding symbols again. This happens again and again. Unbeknownst to you, there is a Chinese speaker outside of the room is passing you these slips. The writing on the slips is a conversation being carried out, and what you are writing are perfectly fine responses to the parts of the conversations initiated by the Chinese speaker. The question that Searle asks is if you, the person inside the room, understand Chinese? To the Chinese speaker outside the room you must, because you are producing perfectly fine responses. But, Searle, and many other people, say that you do not understand Chinese, because you are just parroting back what is in the book. The argument is that this is what computers do, they just follow their programming, with no real understanding of their actions, just like the bad guys in Uncharted and the computer opponent in Video Olympics.

What about Deep Blue and AlphaGo, do they understand what they are doing? Here you might get some disagreement, particularly about AlphaGo. But, probably most people would end up saying no, they don't really understand what they are doing, and if they do, it's just about the one small thing they are good at. There are some counter-arguments to Searle's Chinese Room Problem. Some people say that if you consider the whole system, not just the person inside the room, but also the book, the room, everything, that the whole system understands Chinese at some level. One of my dissertation advisors, and one of the greats in psychology and computer science, was Herbert Simon. His answer to the Chinese Room was to give the person a window. That by being able to make connections between the symbols coming in and what was happening in the outside world, that would drive understanding. That dividing line between rote repetition of instructions and understanding is not a clear one. Once we more fully comprehend what it means to understand, then we will be better able to understand what it means to be human.

Will we ever create an artificial intelligence? Like any good professor, I can argue on both sides of that issue. I can appreciate Searle's argument, that computers follow their programming and nothing more. Scientists may be able to break us down into our constituent parts, but there's something special about how those parts are arranged that preclude putting them back together again. But, I can also argue that we humans are just machines, made of flesh and bone and not silicon, following our own programming. There have been for some time programs that learn and modify their own programming. That's the heart of what neural networks do, modify the connections between their neuron-like units. At some point we'll be able to model that behavior within our computer programs.

To close our philosophical thoughts here, I have been reading a book called The Innovators by Walter Isaacson. In addition to tracing this question of can computers think, he also stresses that innovation usually comes from bringing people with different strengths together, like Steve Jobs and Steve Wozniak or Bill Gates and Paul Allen. In one of the middle chapters Issaacson writes about the creation of the internet, and the role of Joseph Licklider, a professor at MIT. In 1960 Licklider wrote a paper called "Man-Computer Symbiosis" in which he states, quote "The hope is that, in not too many years, human brains and computing machines will be coupled together very tightly, and that the resulting partnership will think as no human brain has ever thought and process data in a way not approached by the information-handling machines we know today." End quote. Perhaps the role of an artificial intelligence is not to mimic human intelligence, but rather to complement it and allow us to consider ideas and solutions we wouldn't otherwise. Indeed, in looking at the commentary across the 5 games that AlphaGo played against Lee Sodol, one sees that happening. In the second game, move 37, AlphaGo made a move that surprised Sodol and the commentators. It was only after some reflection how that move actually did work to AlphaGo's advantage. A human Go master wouldn't have made that move, but it worked with AlphaGo's strategy. Here is a case of a computer program seemingly being creative and allowing humans to see different patterns.

Okay, this ends another episode of the Cognitive Gamer podcast. I enjoy thinking and reading about these issues involving what it means to be intelligent. Next time, we will consider decision making in game playing. Between now and then, if you have any questions or comments, please email me at steve@cognitivegamer.com. I would love to hear from you. Also, be sure to like my facebook page, Cognitive Gamer, and to visit the website cognitivegamer.com. You can also follow me on Twitter, at cognitive underscore gamer. Until next time, remember to think about what you play, and have fun doing it.