Will Computers Outsmart Mathematicians?

Kevin Buzzard

20 January 2021

Computers V Humans

Welcome to everybody, and thanks for coming! In this talk, I will compare the skills of a computer with the skills of a human. Both sides have abilities which the other lacks. The question we will ultimately consider: which of them will be better at mathematics? Of course, we will have to say what we *mean* by "mathematics" here.

What is a Computer?

Facts about computers:

- A computer is a device which can run a computer program.
- A computer program is a list of logical instructions. The computer follows the instructions unthinkingly, but quickly.
- You can think of a computer as an *unimaginative* but *efficient* assistant.
- A computer can work out 34873847634 × 2847238746 much quicker than a human.

But can a computer *think*? Can it come up with a proof of Pythagoras' theorem? Are those two questions related?

Memory (RAM)

Computer Memory: a bunch of on/off switches.
In the 1940s and 1950s, the switches were valves.

Modern Memory

After the silicon revolution, switches became transistors.

Advantages of transistors:

- Smaller
- Faster
- Less likely to break

A modern phone contains billions of these switches. Nowadays, a computer can switch many millions of switches in one second.

What Use is a Switch

A switch in a computer can be *on* or *off*. Hence a switch can be used to represent "0/1". Using binary notation, 0s and 1s can be used to encode larger numbers. Numbers can be used to encode letters (e.g. "A=1, B=2, C=3, . . . "), and much more. Computer programs can hence analyse systems which can be described with a finite amount of data.

Chess

Chess is a game, played on an 8 × 8 board, with a finite list of rules. Goes back over 1000 years. Human players play by having *ideas*, and being *creative*. In the 1970s, computers were taught to play chess. Computers played by trying every move and response, quickly. Computers were not very good, at first. Then IBM got involved. By the late 1990s, computers were beating the best human grandmasters. How did this happen?

Computers Playing Chess

Two main reasons:

1) Computers were getting really fast by the 1990s.
2) IBM trained their chess program, *Deep Blue*, on hundreds and thousands of Grandmaster games. Computers took human ideas, and turned them into instructions.

This area of computer science is called *artificial intelligence*, or AI. Nowadays, humans use computers to help them to learn about chess.

GO

The board game *go* is played on a much bigger board. Go is a game, played on a 19 × 19 board, with a finite list of rules. Computer scientists taught these rules to computers. The first computer programs were not very good. Then DeepMind got involved. By the late 2010s, computers were beating the best human grandmasters. How did this happen?

AI Getting Better

DeepMind's program, *alpha go*, used more advanced *machine learning* techniques. Machine learning is a part of modern AI research.

Open question: Is this program "having ideas"? Or is it just trying everything, very quickly?

After DeepMind's success, some computer scientists declared that the domain of board games was "solved".

What is Mathematics?

Mathematics is all around us. It is the language of physics, engineering, computer science, finance.

Pure mathematics is however not application-driven. Today, pure mathematics is about stating and proving abstract *theorems*. A *theorem* is an abstract mathematical statement, like a logic puzzle. Proving the theorem is like solving the puzzle. Mathematical theorem proving is a game.

The Game of Pure Mathematics

Mathematic theorem proving is a game.
Pure mathematics has a finite list of rules. Mathematics is, however, played on an *infinite board*.

$\{0, 1, 2, 3, \ldots\}$

It is hence a natural target for AI, after board games like chess or Go.

What is Pure Mathematics?

Pure mathematics started off as an *abstraction* of the physical world.

Two pebbles, and two more pebbles, make four pebbles.
Two sheep, and two more sheep, make four sheep.
The *underlying abstraction*: 2 + 2 = 4.
Is there such a thing as "2"?

Pythagoras' Theorem

Pythagoras' theorem $A + B = C$ is also an abstraction. It is about an "abstract square".

The Rules of Shapes

Euclid's Elements was written over 2000 years ago. Euclid begins Book 1 by writing down the rules – five "postulates" and five "common notions". By the end of book 1 he has proved Pythagoras' Theorem.

The Rules of Numbers

Let's talk about the rules which define numbers.
Let's stick to the so-called *natural numbers* {0, 1, 2, 3,...}.
What is a *finite* list of rules which uniquely characterise these numbers?

Peano's Rules for Natural Numbers

Natural numbers: {0, 1, 2, 3, 4,...}.

The 19th century Italian mathematician Giuseppe Peano wrote down rules which characterise them.

Peano's rules:

- 0 is a number;
- If n is a number, then the *successor* $S(n)$ of n (that is, the number after n) is a number;
- That's it.

The Birth of Number

Peano's rules:

- 0 is a number;
- If n is a number, then the *successor* $S(n)$ of n is a number;
- That's it.

The *successor* of n is the number after n. For example the successor of 37 is 38.

We will use the notation $S(n)$ for the number after n. Why don't we just call it $n + 1$? Because we have not yet *defined* addition of numbers! Before we learn to add, we must first learn to count.

Most humans do not usually consider mathematics in this primitive state.

Lean

Lean is free open-source software being developed by Microsoft Research. Lean is a computer program which knows the rules of logic, and can hence be taught logical games and puzzles.

Example: pure mathematics. Lean is an *interactive theorem prover*. Other examples of interactive theorem provers: Coq, Agda, Isabelle, Metamath, HOL Light, HOL 4, . . .

Let's teach Peano's axioms to Lean.

Peano's Axioms

- 0 is a number;
- If n is a number, then the *successor* $S(n)$ of n is a number;
- That's it.

Consequence of "that's it": every number is built using the first two rules.

So say you want to do something with numbers.

- If you've done it for 0,. . .
- . . . and if you've done it for n, then you've done it for $S(n)$,. . .
- . . . then you've done it for all the numbers.

The Principle of Mathematical Induction

- If you've done it for 0,. . .
- . . . and if you've done it for n, then you've done it for $S(n)$,. . .
- . . . then you've done it for all the numbers.

Remember, numbers have *just been born* at this point.

If we want to add numbers, we must first define addition using only the principles we already have.

For example, let's try and define how to add 2 to a number.

Defining Addition

- If you've done it for 0,
- and if you've done it for n, then you've done it for $S(n)$,
- then you've done it for all the numbers.

First we need to define 2 + 0.
Let's define 2 + 0 to be 2.
Then 2 + 1 should be the number after 2.
And 2 + 2 should be the number after that.

In general, $2 + S(n)$, should be the number after $2 + n$.
So if we've defined $2 + n$, we can define $2 + S(n)$.
The formula: $2 + S(n) = S(2 + n)$.
Let's teach that to the computer.

Now let's play a game! Let's show that 2 + 2 = 4, using only the rules.
Reminder of the definitions:

- $1 = S(0)$, $2 = S(1)$, $3 = S(2)$, $4 = S(3)$.
- $2 + 0 = 2$;
- $2 + S(n) = S(2 + n)$.

So…

2 + 2 = 2 + *S*(1) (by definition of 2)
= *S*(2 + 1) (by definition of +)
= S(2 + S(0)) (by definition of 1)
= *S*(*S*(2 + 0)) (by definition of +)
= *S*(*S*(2)) (by definition of +)
= *S*(3) (by definition of 3)
= 4 (by definition of 4).
So 2 + 2 = 4. QED!

<u>2+2=4</u>
Let's see if Lean can do it.

<u>QED</u>
We taught the computer the rules, and some basic definitions.
An AI then solved the "2 + 2 = 4" level of this maths game.
Of course, a basic *calculator* could solve this easily!
The difference: a calculator has been *told a method for working out* 2 + 2. Lean has *worked it out by itself*.

<u>How Much Further Does This Go?</u>
So, what if we teach the computer an entire undergraduate pure mathematics degree?

A team of people across the world are working on this right now. Including undergraduates at my university and beyond. It takes time. But it is *gamifying mathematics*. Students find it *fun*.

AI experts have begun (this year!) to use machine learning techniques on our fledgeling database (Stanislas Polu, OpenAI). They can prove basic results about natural numbers, and a lot more. Other research groups are trying the same with other systems.

<u>The Future</u>
This is just a research project right now. Perhaps our current attempts will end up like those early lousy chess computers. But the idea is *in principle* very appealing. And one day, somebody will get it to work. And then we will have a system which can try to pass an undergraduate mathematics degree. And then we can do more.

When will computers start to beat undergraduates at proving theorems? My belief: within my lifetime.

Thank you for coming!