Dwarkesh Podcast #75 - Francois Chollet, Mike Knoop - LLMs won't lead to AGI - $1,000,000 Prize to find true solution

Published - June 11, 2024

Transcribed by - thepodtranscripts.com

Dwarkesh Patel
Today I have the pleasure to speak with François Chollet, who is an AI researcher at Google and creator of Keras. He's launching a prize in collaboration with Mike Knoop, the co-founder of Zapier, whom we'll also be talking to in a second. It's a million dollar prize to solve the ARC benchmark that he created.

First question, what is the ARC benchmark? Why do you even need this prize? Why won't the biggest LLM we have in a year be able to just saturate it?

François Chollet
ARC is intended as a kind of IQ test for machine intelligence. What makes it different from most LLM benchmarks out there is that it's designed to be resistant to memorization. The way LLMs work is that they're basically this big interpolative memory. The way you scale up their capabilities is by trying to cram as much knowledge and patterns as possible into them.

By contrast, ARC does not require a lot of knowledge at all. It's designed to only require what's known as core knowledge. It's basic knowledge about things like elementary physics, objectness, counting, that sort of thing. It's the sort of knowledge that any four-year-old or five-year-old possesses.

What's interesting is that each puzzle in ARC is novel. It's something that you've probably not encountered before, even if you've memorized the entire internet. That's what makes ARC challenging for LLMs. So far, LLMs have not been doing very well on it. In fact, the approaches that are working well are more towards discrete program search, program synthesis.

Dwarkesh Patel
First of all, I'll make a comment that I'm glad that as a skeptic of LLM, you have yourself put out a benchmark. Is it accurate to say that if the biggest model we have in a year is able to get 80% on this, then your view would be that we are on track to get AGI with LLMs? How would you think about that?

François Chollet
I'm pretty skeptical that we're going to see an LLM do 80% in a year. That said, if we do see it, you would also have to look at how this was achieved. If you just train the model on millions or billions of puzzles similar to ARC, you're relying on the ability to have some overlap between the tasks that you train on and the tasks that you're going to see at test time. You're still using memorization.

Maybe it can work. Hopefully, ARC is going to be good enough that it's going to be resistant to this sort of brute force attempt but you never know. Maybe it could happen. I'm not saying it’s not going to happen. ARC is not a perfect benchmark. Maybe it has flaws. Maybe it could be hacked in that way.

Dwarkesh Patel
What would GPT-5 have to do so that you would be very confident that it’s on the path to AGI?

François Chollet
This is what would make me change my mind about LLMs. I would need to start seeing a critical mass of cases where you show the model something it has not seen before — a task that's truly novel from the perspective of its training data — and it can actually adapt on the fly.

This is true for LLMs but really this would catch my attention for any AI technique out there. If I can see the ability to adapt to novelty on the fly and pick up new skills efficiently, then I would be extremely interested. I would think this is on the path to AGI.

Dwarkesh Patel
The advantage they have is that they do get to see everything. Maybe I'll take issue with how much they are relying on that, but obviously they're relying on that more than humans do. They do have so much in distribution, to the extent that we have trouble distinguishing whether an example is in distribution or not.

If they have everything in distribution, then they can do everything that we can do. Maybe it's not in distribution for us. Why is it so crucial that it has to be out of distribution for them? Why can't we just leverage the fact that they do get to see everything?

François Chollet
Basically you’re asking what's the difference between actual intelligence — the ability to adapt to things you've not been prepared for — and pure memorization, like reciting what you've seen before.

It's not just some semantic difference. The big difference is that you can never pre-train on everything that you might see at test time because the world changes all the time. It's not just the fact that the space of possible tasks is infinite. If you're trained on millions of them, you've only seen zero percent of the total space. It's also the fact that the world is changing every day.

This is why we, the human species, have developed intelligence in the first place. If there was such a thing as a distribution for the world — for the universe, for our lives — then we would not need intelligence at all. In fact, many creatures, many insects for instance, do not have intelligence. Instead they have hardcoded programs in their connectomes, in their genes, behavioral programs that map some stimuli to appropriate responses. They can actually navigate their lives and their environment in a way that's very evolutionarily fit without needing to learn anything.

If our environment were static and predictable enough, what would have happened is that evolution would have found the perfect behavioral program: a hard-coded, static behavioral program. It would have written it into our genes. We would have a hard-coded brain connectome. That's what we would be running on. But that's not what happened.

Instead, we have general intelligence. We are born with extremely little knowledge about the world. We are born with the ability to learn very efficiently and to adapt in the face of things that we've never seen before. That's what makes us unique. That's what is really, really challenging to recreate in machines.

Dwarkesh Patel
Before we dive deeper into that, I'm going to overlay some examples of what an ARC-like challenge looks like for the YouTube audience. For people listening on audio, can you describe what a sample ARC challenge would look like?

François Chollet
One ARC puzzle looks kind of like an IQ test puzzle. You have a number of demonstration input-output pairs. One pair is made up of two grids. One grid shows you an input, and the second grid shows you what you should produce as a response to that input.

You get a couple pairs like this to demonstrate the nature of the task and what you're supposed to do with your inputs. You then get a new test input. Your job is to produce the corresponding test output. You look at the demonstration pairs and from that you figure out what you're supposed to do. You show that you've understood it on this new test pair.

Importantly, the knowledge basis you need to approach these challenges is just core knowledge. It includes basic concepts like what makes an object, counting, geometry, topology, symmetries, etc. It's extremely basic knowledge. LLMs for sure possess such knowledge. Any child possesses such knowledge.

What's really interesting is that each puzzle is new. It's not something you'll find elsewhere on the internet. Whether you're a human or a machine, you have to approach every puzzle from scratch and reason your way through it. You can't just fetch the response from memory.

Dwarkesh Patel
One contention here is that we are only now getting multimodal models that are trained to do spatial reasoning due to the data they're trained on. Whereas not only humans but our

ancestors have had to learn over billions of years of evolution how to understand abstract physical and spatial properties and recognize patterns there.

One view is that in the next year, as we gain models that are natively multimodal capability rather than as an add-on, they will understand these kinds of patterns because that's something we 'd see natively. Right now, ARC sees a JSON string of 100100 and is supposed to recognize a pattern there. Even if you showed a human a sequence of these numbers, they would have a challenge making sense of the question you're asking.

Why wouldn't multimodal models, which we're on the path to unlocking right now, be so much better at ARC-type spatial reasoning as soon as we get them?

François Chollet
That's an empirical question. I guess we'll see the answer within a few months. My response is that our grids are just discrete 2D grids of symbols and are pretty small. If you flatten an image as a sequence of pixels for example, you get something that's actually very difficult to parse.

That's not true for ARC because the grids are very small. You only have 10 possible symbols, They are 2D grids that are actually very easy to flatten as sequences. Transformers, LLMs, are very good at processing sequences.

In fact, you can show that LLMs do fine with processing ARC-like data by simply fine-tuning an LLM on subsets of the tasks and then testing it on small variations of these tasks. You'll see that the LLM can encode solution programs just fine for tasks it has seen before. It doesn't really have a problem parsing the input or figuring out the program.

The reason LLMs don't do well on ARC is really just the unfamiliarity aspect. Each new task is different from every other task. You cannot memorize the solution programs in advance. You have to synthesize a new solution program on the fly for each new task. That's really what LLMs are struggling with.

Dwarkesh Patel
Before I play more devil's advocate, I just want to step back and explain why I'm especially interested in having this conversation. Obviously there's the million dollar ARC Prize and I'm excited to play around with it myself.

The Vesuvius Challenge was Nat Friedman's prize for decoding scrolls from the Herculaneum library that were buried in the volcano. The winner of that was a 22-year-old who was listening to this podcast, Luke Farritor. Hopefully somebody listening to this will find this challenge intriguing and find a solution.

Dwarkesh Patel
I've recently had on a lot of people who are bullish on LLMs. I've had discussions with them before interviewing you about how we explain the fact that LLMs don't seem to be natively performing that well on ARC.

I found their explanations somewhat contrived. I'll try out some of their reasons on you. It is actually an intriguing fact that some of these problems are relatively straightforward for humans to understand, yet the models struggle with them if you just input them natively.

François Chollet
All of them are very easy for humans. Any smart human should be able to do 90-95% on ARC. Even a five-year-old with very, very little knowledge could definitely do over 50%.

Dwarkesh Patel
I agree that smart humans will do very well on this test, but the average human will probably be mediocre.

François Chollet
Not really, we actually tried with average humans. They scored about 85.

Dwarkesh Patel
That was with Amazon Mechanical Turk workers, right? I honestly don't know the demographic profile of Amazon Mechanical Turk workers. Imagining them interacting with Amazon's remote work platform, I'm guessing that's not the median human across the planet.

The broader point here is that we see the spectrum in humans and humans obviously have AGI. But even within humans you see a spectrum where some people are relatively dumber. They'll perform worse on IQ-like tests.

For example, there's Raven's Progressive Matrices. Look at how the average person performs on that. If you look at the kind of questions that are hit or miss — half of people will get it right, half of people will get it wrong — we might think they're kind of trivial.

Humans have AGI but from relatively small tweaks, you can go from somebody who misses these kinds of basic IQ test questions to somebody who gets them all right. We'll talk about some of the previous performances that people have tried with these models.

Jack Cole with a 240 million parameter model got 35%. Doesn't that suggest that they're on this spectrum that clearly exists within humans, and they're going to be saturated pretty soon?

François Chollet
There's a bunch of interesting points here. There is indeed a branch of LLM approaches spearheaded by Jack Cole that are doing quite well. They are state-of-the art in fact.. But you have to look at what's going on there. There are two things.

The first thing is that to get these numbers, you need to pre-train your LLM on millions of generated ARC tasks. Of course, compare that to a five-year-old child looking at ARC for the first time. The child has never done an IQ test before and has never seen something like an ARC test before.

The only overlap between what they know and what they have to do in the test is core knowledge. It’s knowing about counting, objects, symmetries, etc. They're still going to do really well. They're going to do much better than the LLM trained on millions of similar tasks.

There’s a second thing to note about the Jack Cole approach. One thing that's really critical to making the model work at all is test time fine-tuning. By the way, that's something that's really missing from LLM approaches right now. Most of the time when you're using an LLM, it's just doing static inference. The model is frozen. You're just prompting it and getting an answer. The model is not actually learning anything on the fly. Its state is not adapting to the task at hand.

What Jack Cole is actually doing is that for every test problem, it’s on-the-fly fine-tuning a version of the LLM for that task. That's really what's unlocking performance. If you don't do that, you get like 1-2%, something completely negligible. If you do test time fine-tuning and you add a bunch of tricks on top, then you end up with interesting performance numbers.

What it's doing is trying to address one of the key limitations of LLMs today: the lack of active inference. It's actually adding active inference to LLMs. That's working extremely well, actually. So that's fascinating to me.

Dwarkesh Patel
There are so many interesting rabbit holes there. A lot of the scale maximalists share your broader perspective that you need to unlock the adaptive/test time compute. They think that in addition to scaling, you need things like adaptive compute or some sort of RL to get the System 2 working. Their perspective is that this is a relatively straightforward thing that will be added atop the representations that a scaled up model has greater access to.

François Chollet
It's not just a technical detail. It's not a straightforward thing. It is everything. It is the important part. The scale maximalists refer to scaling laws, which are the empirical relationship that you can draw between how much compute you spend on training a model and the performance you're getting on benchmark.

Of course the key question here is, how do you measure performance? What is it that you're actually improving by adding more compute and more data? It's benchmark performance.

The way you measure performance is not a technical detail. It's not an afterthought because it's going to narrow down the set of questions that you're asking. Accordingly, it's going to narrow down the set of answers that you're looking for.

If you look at the benchmarks we are using for LLMs, they are all memorization-based benchmarks. Sometimes they are literally just knowledge-based, like a school test. Even if you look at the ones that are explicitly about reasoning, if you look closely you realize that in order to solve them, it's enough to memorize a finite set of reasoning patterns. You just reapply them. They're like static programs.

LLMs are very good at memorizing small static programs. They've got this sort of bank of solution programs. When you give them a new puzzle, they can just fetch the appropriate program and apply it. It looks like reasoning but it's not really doing any sort of on-the-fly program synthesis. All it's doing is program fetching.

You can actually solve all these benchmarks with memorization. If you look at the models and what you're scaling up here, they are big parametric curves fitted to a data distribution. They're basically these big interpolative databases, interpolative memories. Of course, if you scale up the size of your database and cram more knowledge and patterns into it, you are going to be increasing its performance as measured by a memorization benchmark.

That's kind of obvious. But as you're doing it, you are not increasing the intelligence of the system one bit. You are increasing the skill of the system. You are increasing its usefulness, its scope of applicability, but not its intelligence because skill is not intelligence. That's the fundamental confusion that people run into. They're confusing skill and intelligence.

Dwarkesh Patel
There are a lot of fascinating things to talk about here: skill, intelligence, interpolation. Let’s talk about the point that they’re fitting some manifold that maps the input data. A reductionist way to talk about the human brain is that it's just axons firing at each other. But we don't care about the reductionist explanation. We care about what happens at the macroscopic level when these things combine.

As far as interpolation goes, let's look at one of the benchmarks. There's a benchmark that does grade school math. These are problems that a smart high schooler would be able to solve. It's called GSM8K. These models get 95% on it. Basically, they always nail it.

François Chollet
Sure, that's a memorization benchmark.

Dwarkesh Patel
Let's talk about what that means. Here's one question from that benchmark:

"30 students are in a class. One-fifth of them are 12-year-olds, One-third are 13-year-olds, One-tenth are 11-year-olds. How many of them are not 11, 12, or 13 years old?

I agree this is not rocket science. You can write down on paper how you go through this problem. A smart high school kid should be able to solve it. About memorization, it still has to reason through how to think about fractions, the context of the whole problem, and then combine different calculations to write the final answer.

François Chollet
It depends on how you want to define reasoning. There are two definitions you can use. One is, I have available a set of program templates. It's the structure of the puzzle, which can also generate its solution. I'm going to identify the right template, which is in my memory, input the new values into the template, run the program, and get the solution. You could say this is reasoning. I say, "yeah sure, okay."

Here's another definition of reasoning. When you're faced with a puzzle and you don't already have a program in memory to solve it, it's the ability to synthesize on the fly a new program based on bits and pieces of existing programs that you have. You have to do on-the-fly program synthesis. That's actually dramatically harder than just fetching the right memorized program and reapplying it.

Dwarkesh Patel
Maybe we are overestimating the extent to which humans are so sample efficient. They also need training in this way. They have to drill in these pathways of reasoning through certain kinds of problems.

Let's take math, for example. It's not like you can just show a baby the axioms of set theory and now they know math. When they're growing up, you have to teach them years of pre-algebra. Then you have a year of teaching them drills and going through the same kind of problem in algebra, then geometry, pre-calculus, calculus.

Isn't that like the same kind of thing? You can't just see one example and now you have the program. You actually have to drill it. These models also had to drill it with a bunch of pre-training data.

François Chollet
Sure. In order to do on-the-fly program synthesis, you actually need building blocks to work from. Knowledge and memory are tremendously important in the process. I'm not saying it's memory vs. reasoning. In order to do effective reasoning, you need memory.

Dwarkesh Patel
But it sounds compatible with your story. Through seeing a lot of different kinds of examples, these things can learn to reason within the context of those examples. We can also see it within bigger and bigger models.

That was an example of a high school-level math problem. Let's say a model that's smaller than GPT-3 couldn't do that at all. As these models get bigger, they seem to be able to pick up bigger and bigger patterns.

François Chollet
It's not really a size issue. It's more like a training data issue in this case.

Dwarkesh Patel
Well, bigger models can pick up these kinds of circuits. Smaller models apparently don't do a good job of doing that even if you were to train them on this kind of data. Doesn't that just suggest that as you have bigger and bigger models, they can pick up bigger and bigger pathways or more general ways of reasoning?

François Chollet
Absolutely.

Dwarkesh Patel
But then isn't that intelligence?

François Chollet
No, it's not. If you scale up your database and keep adding more knowledge and program templates to it, then sure it becomes more and more skillful. You can apply it to more and more tasks. But general intelligence is not task-specific skill scaled up to many skills, because there is an infinite space of possible skills.

General intelligence is the ability to approach any problem, any skill, and very quickly master it using very little data. This is what makes you able to face anything you might ever encounter. This is the definition of generality. Generality is not specificity scaled up. It is the ability to apply your mind to anything at all, to arbitrary things. This fundamentally requires the ability to adapt, to learn on the fly efficiently.

Dwarkesh Patel
My claim is that by doing pre-training on bigger and bigger models, you are gaining that capacity to generalize very efficiently. Let me give you an example. Your own company Google, in their paper on Gemini 1.5, had this very interesting example. They would give the model, in context, the grammar book and the dictionary of a language that has fewer than 200 living speakers. It's not in the pre-training data. You just give it the dictionary and it basically is able to speak this language and translate to it, including the complex and organic ways in which languages are structured.

If you showed me a dictionary from English to Spanish, I'm not going to be able to pick up how to structure sentences and how to say things in Spanish. Because of the representations that it has gained through this pre-training, it is able to now learn a new language extremely efficiently. Doesn't that show that this kind of pre-training actually does increase your ability to learn new tasks?

François Chollet
If you were right, LLMs would do really well on ARC puzzles because ARC puzzles are not complex. Each one of them requires very little knowledge. Each one of them is very low on complexity. You don't need to think very hard about it. They're actually extremely obvious for human

Even children can do them but LLMs cannot. Even LLMs that have 100,000x more knowledge than you do still cannot. The only thing that makes ARC special is that it was designed with this intent to resist memorization. This is the only thing. This is the huge blocker for LLM performance.

If you look at LLMs closely, it's pretty obvious that they're not really synthesizing new programs on the fly to solve the task that they're faced with. They're very much reapplying things that they've stored in memory. For instance, one thing that's very striking is that LLMs can solve a Caesar cipher, transposing letters to code a message. That’s a very complex algorithm, but it comes up quite a bit on the internet. They've basically memorized it.

What's really interesting is that they can do it for a transposition length of like three or five, because those are very common numbers in examples provided on the internet. If you try to do it with an arbitrary number like nine, it's going to fail. It does not encode the generalized form of the algorithm, but only specific cases. It has memorized specific cases of the algorithm. If it could actually synthesize on the fly the solver algorithm, then the value of n would not matter at all, because it does not increase the problem complexity.

Dwarkesh Patel
I think this is true of humans as well.

François Chollet
Humans use memorization pattern matching all the time, of course, but humans are not limited to memorization pattern matching. They have this very unique ability to adapt to new situations on the fly. This is exactly what enables you to navigate every new day in your life.

Dwarkesh Patel
There was some study that chess grandmasters will perform very well within the context of the moves that—

François Chollet
That's an excellent example because chess, at the highest level, is all about memorization, chess memorization.

Dwarkesh Patel
What is your explanation for the original question of why Gemini 1.5 was able, in context, to learn a language, including the complex grammar structure? Doesn't that show that they can pick up new knowledge?

François Chollet
I would assume that it has simply mined from its extremely extensive, unimaginably vast training data. It has mined the required template and then it's just reusing it. We know that LLMs have a very poor ability to synthesize new program templates like this on the fly or even adapt existing ones. They're very much limited to fetching.

Dwarkesh Patel
Suppose there's a programmer at Google. They go into the office in the morning. At what point are they doing something that 100% cannot be due to fetching some template?

Suppose they were an LLM. What could they not do if they had only fetched some template from their program? At what point do they have to use this so-called extreme generalization capability?

François Chollet
Forget about Google software developers. For every human, every day of their lives is full of novel things that they've not been prepared for. You cannot navigate your life based on memorization alone. It's impossible.

Dwarkesh Patel
It seems like you also agree they're not doing just "memorization." It seems like you're saying they're less capable of generalization. I'm just curious about the kind of generalization they do.

If you get into the office and you try to do this kind of generalization, you're going to fail at your job. Let’s say you're a programmer. What is the first point when you try to do that kind of generalization, you would lose your job because you can't do the extreme generalization?

François Chollet
Take this situation, for instance. You've never been here in this room. Maybe you've been in this city a few times. There's a fair amount of novelty. You've never been interviewing me. There's a fair amount of novelty in every hour of every day in your life. By and large, it’s in fact more novelty than any LLM could handle. If you just put an LLM in a robot, it could not be doing all the things that you've been doing today.

Take self-driving cars, for instance. You take a self-driving car operating in the Bay Area. Do you think you could just drop it in New York City or drop it in London, where people drive on the left? No, it's going to fail. Not only can it not generalize to a change in driving rules, but you cannot even make it generalize to a new city. It needs to be trained on each specific environment.

Dwarkesh Patel
I agree that self-driving cars aren't AGI.

François Chollet
But it's the same type of model. They're transformers as well. It's the same architecture.

Dwarkesh Patel
I don’t know. Apes also have brains with neurons in them, but they're less intelligent because they're smaller.

We can get into that. I still don't understand this concrete thing. We also need training. That's why education exists. That's why we had to spend the first 18 years of our life doing drills.

François Chollet
We have a memory, but we are not a memory. We are not limited to just a memory.

Dwarkesh Patel
I’m denying the premise that that's the only thing these models are necessarily doing. Suppose you just subbed out a remote work with an LLM and they're a programmer. What is the first point at which you realize this is not a human, this is an LLM?

François Chollet
How about I just send them an ARC puzzle and see how they do?

Dwarkesh Patel
No, like part of their job.

François Chollet
You have to deal with novelty all the time.

Dwarkesh Patel
Is there a world in which all the programmers are replaced and we're still saying, "ah, but they're only doing memorization-laden programming tasks." In that world, are they still producing a trillion dollars worth of output in the form of code?

François Chollet
Software development is actually a pretty good example of a job where you're dealing with novelty all the time. If you're not, I'm not sure what you're doing.

I personally use generative AI very little in my software development job. Before LLMs, I was also using Stack Overflow very little. Some people maybe are just copy-pasting stuff from Stack Overflow, or nowadays copy-pasting stuff from an LLM.

Personally, I try to focus on problem-solving. The syntax is just a technical detail. What's really important is problem-solving. The essence of programming is engineering mental models and mental representations of the problem you're trying to solve.

Dwarkesh Patel
We have many people who can interact with these systems themselves. You can go to ChatGPT and say, "here's a specification of the kind of program I want." They'll build it for you.

François Chollet
As long as there are many examples of this program on GitHub, Stack Overflow, and so on, sure they will fetch the program for you from their memory.

Dwarkesh Patel
But you can change arbitrary details. You can say, "I need it to work on this different kind of server."

François Chollet
If that were true, there would be no software engineers today.

Dwarkesh Patel
I agree we're not at a full AGI yet. These models have fewer than a trillion parameters. A human brain has somewhere on the order of 10-30 trillion synapses. If you were just doing

some naive math, you're at least 10x underparameterized. I agree we're not there yet, but I'm confused about why we're not on the spectrum.

Yes, I agree that there are many kinds of generalization they can't do. But it seems like they're on this kind of smooth spectrum that we see even within humans. Some humans would have a hard time doing an ARC-type test. We see that based on the performance on Raven's progressive matrices-type IQ tests.

François Chollet
I'm not a fan of IQ tests because, for the most part, you can train on IQ tests and get better at them. They're very much memorization-based. This is actually the main pitfall that ARC tries not to fall for.

Dwarkesh Patel
Let’s say all remote jobs are automated in the next five years. I mean at least the remote jobs that don't require you to be a sort of a service, like a salesperson, where you want the human to be talking. I mean more like programming.

In that world, would you say that that's not possible because a programmer needs to do many things that definitely require things that would not be in any pre-training corpus?

François Chollet
Sure. In five years, there will be more software engineers than there are today, not fewer.

Dwarkesh Patel
I'm still not sure. I studied computer science. If I had become a code monkey out of college, what would I be doing? I go to my job. My boss tells me to do something? When does he realize I'm an LLM, if I were an LLM?

François Chollet
Probably on the first day. Again, if it were true that LLMs could generalize to novel problems like this — actually develop software to solve a problem they've never seen before — you would not need software engineers anymore.

If I look at how people are using LLMs in their software engineering job today, they're using it as a Stack Overflow replacement. They're using it as a way to copy-paste code snippets to perform very common actions. What they actually need is a database of code snippets. They don't actually need any of the abilities that actually make them software engineers.

Dwarkesh Patel
Let's step back on interpolation. Why isn't creativity just interpolation in a higher dimension where — if we're going to use the ML language — a bigger model can learn a more complex manifold?

If you read a biography of a scientist, they're not zero-shotting new scientific theories. They're playing with existing ideas. They're trying to juxtapose them in their head. In the tree of intellectual descendants, they try out some slightly different evolutionary path. You sort of run the experiment there in terms of publishing the paper or whatever.

It seems like a similar kind of thing to what humans are doing. There's a higher level of generalization. Bigger and bigger models seem to be approaching higher and higher levels of generalization. GPT-2 couldn't do grade school-level math problems that required more generalization than it had the capability to do. GPT-3 and GPT-4 can.

François Chollet
Not quite. GPT-4 has a higher degree of skill and a higher range of skills. It has the same degree of generalization.

Dwarkesh Patel
I don't want to get into semantics here. Why can't creativity just be interpolation on a higher dimension?

François Chollet
Interpolation can absolutely be creative. To your point, I do think that on some level humans also do a lot of memorization, reciting, pattern matching, and interpolation as well. It's very much a spectrum between pattern matching and true reasoning. Humans are never really at one end of the spectrum. They're never really doing pure pattern matching or pure reasoning. They're usually doing some mixture of both.

This is true even if you're doing something that seems very reasoning-heavy, like proving a mathematical theorem. As you're doing it, you're doing quite a bit of discrete search in your mind and quite a bit of actual reasoning. You're also very much guided by intuition and pattern matching. You're guided by the shape of proofs that you've seen before, by your knowledge of mathematics.

All of our thoughts, everything we do, is a mixture of interpolated memorization-based thinking, Type 1 thinking, and Type 2 thinking.

Dwarkesh Patel
Why are bigger models more sample efficient?

François Chollet
Because they have more reusable building blocks that they can lean on to pick up new patterns in their training data.

Dwarkesh Patel
Does that pattern keep continuing as you keep getting bigger and bigger?

François Chollet
It does to the extent that the new patterns you're giving the model to learn are a good match for what it has learned before. If you present something that's actually novel that is not in a steady distribution, like an ARC puzzle for instance, it will fail.

Dwarkesh Patel
Let me make this claim. The program synthesis is a very useful intuition pump. Why can't this be the case for what's happening in the transformer?

The early layers are figuring out how to represent the inputting tokens. The middle layers do this kind of program search, program synthesis, and they combine the inputs to all the circuits in the model. They go from the low-level representation to a higher-level representation near the middle of the model. They use these programs. They combine these concepts. What comes out the other end is the reasoning based on that high-level intelligence.

François Chollet
Possibly. Why not? But if these models were actually capable of synthesizing novel programs, however simple, they should be able to do ARC. Because for any ARC task, if you write down the solution program in Python, it's not a complex program. It's extremely simple. Humans can figure it out. Why can't LLMs do it?

Dwarkesh Patel
That's a fair point. To turn the question around to you, suppose it's the case that in a year a multimodal model can solve ARC. Let's say it gets 80% or whatever the average human would get. Are we then on track for AGI?

François Chollet
Quite possibly, yes. Honestly, what I would like to see is an LLM-type model solving ARC at 80%, but after having only been trained on core knowledge-related stuff.

Dwarkesh Patel
But human kids, we're necessarily just trained on what we have in our genes…

François Chollet
Let me rephrase that. I want it to be only trained on information that is not explicitly trying to anticipate what's going to be in the ARC test set.

Dwarkesh Patel
Isn't the whole point of ARC that you can't? It's a new type of intelligence test every single time?

François Chollet
Yes, that is the point. If ARC were a perfect, flawless benchmark, it would be impossible to anticipate what's in the test set. ARC was released more than four years ago and so far it's been resistant to memorization. It has, to some extent, passed the test of time. But it's not perfect.

Let's say you try to make by hand hundreds of thousands of ARC tasks. You try to multiply them by programmatically generating variations. You end up with maybe hundreds of millions of tasks. Just by brute forcing the task space, there will be enough overlap between what you're trained on and what's in the test set that you can actually score very highly. With enough scale, you can always cheat.

Dwarkesh Patel
If you can do this for every single thing that supposedly requires intelligence, then what good is intelligence? Apparently, you can just brute force intelligence.

François Chollet
If the world, if your life, were a static distribution then sure, you could just brute force the space of possible behaviors. There are several metaphors for intelligence I like to use. One is that you can think of intelligence as a pathfinding algorithm in future situation space.
I don't know if you're familiar with RTS game development. You have a map, a 2D map, and you have partial information about it. There is some fog of war on your map. There are areas that you haven't explored yet. You know nothing about them. There are also areas that you've explored but you only know what they were like in the past. You don't know how they are like today.

Now, instead of thinking about a 2D map, think about the space of possible future situations that you might encounter and how they're connected to each other. Intelligence is a pathfinding algorithm. Once you set a goal, it will tell you how to get there optimally. Of course, it's constrained by the information you have. It cannot pathfind in an area that you know nothing about. It also cannot anticipate changes.

If you had complete information about the map, then you could solve the pathfinding problem by simply memorizing every possible path, every mapping from point A to point B.

You could solve the problem with pure memory. The reason you cannot do that in real life is because you don't actually know what's going to happen in the future. Life is ever changing.

Dwarkesh Patel
I feel like you're using words like “memorization,” which we would never use for human children. If your kid learns to do algebra and then learns to do calculus, you wouldn't say they've memorized calculus. If they can solve any arbitrary algebraic problem, you wouldn't say they've memorized algebra. You’d say they've learned algebra.

François Chollet
Humans are never really doing pure memorization or pure reasoning.

Dwarkesh Patel
That's only because you're semantically labeling what the human does as skill. But it's a memorization when the exact same skill is done by the LLM, as you can measure by these benchmarks. You can just plug in any sort of math problem.

François Chollet
Sometimes humans are doing the exact same as the LLM is doing. For instance, if you learn to add numbers you're memorizing an algorithm. You're memorizing a program and then you can reapply it. You are not synthesizing on the fly the addition program.

Dwarkesh Patel
Obviously at some point, some human had to figure out how to do addition. A kid doesn’t figure it out by starting from the axioms of set theory and going to how to do addition.

François Chollet
What you learn in school is mostly memorization.

Dwarkesh Patel
My claim is that these models are vastly underparameterized relative to how many flops, how many parameters, you have in the human brain. So it makes sense that they're not going to be coming up with new theorems like the smartest humans can. Most humans can't do that either. What most humans do sounds like something similar to what you are calling memorization, which is memorizing skills or memorizing techniques that you've learned. So it sounds like it's compatible.

Tell me if this is wrong. Is it compatible in your world if all the remote workers are gone but they're doing skills which we can potentially make synthetic data out of? We record every single remote worker's screen. We sort of understand the skills they're performing there. Now we've trained a model that can do all this. All the remote workers are unemployed.

We're generating trillions of dollars of economic activity from AI remote workers. In that world, are we still in the memorization regime?

François Chollet
Sure, with memorization you can automate almost anything as long as it's a static distribution, as long as you don't have to deal with change.

Dwarkesh Patel
Are most jobs part of such a static distribution?

François Chollet
Potentially, there are lots of things that you can automate. LLMs are an excellent tool for automation. But you have to understand that automation is not the same as intelligence. I'm not saying that LLMs are useless. I've been a huge proponent of deep learning for many years.

For many years, I've been saying two things. I've been saying that if you keep scaling up deep learning, it will keep paying off. At the same time I've been saying if you keep scaling up deep learning, this will not lead to AGI.

We can automate more and more things. Yes, this is economically valuable. Yes, potentially there are many jobs you could automate away like this. That would be economically valuable. You're still not going to have intelligence.

So you can ask, what does it matter if we can generate all this economic value? Maybe we don't need intelligence after all. You need intelligence the moment you have to deal with change, novelty, and uncertainty.

As long as you're in a space that can be exactly described in advance, you can just rely on pure memorization. In fact, you can always solve any problem. You can always display arbitrary levels of skills on any task without leveraging any intelligence whatsoever, as long as it is possible to describe the problem and its solution very, very precisely.

Dwarkesh Patel
When they do deal with novelty, then you just call it interpolation.

François Chollet
No, interpolation is not enough to deal with all kinds of novelty. If it were, then LLMs would be AGI.

Dwarkesh Patel
I agree they're not AGI. I'm just trying to figure out if we’re on the path to AGI. The crux here is that it seems to me that these things are on a spectrum and we're clearly covering the earliest part of the spectrum with LLMs.

François Chollet
I think so.

Dwarkesh Patel
Okay, interesting. Here's another thing that I think is evidence for this: grokking.

Clearly, even within deep learning, there's a difference between the memorization regime and the generalization regime. At first they'll just memorize the data set. If you're doing modular addition it’s how to add digits. At some point, if you keep training on that, they'll learn the skill.

The fact that there is that distinction suggests that for the generalized circuit that deep learning can learn, there is a regime where it generalizes if you have an overparameterized model. We don't have that in comparison to all the tasks we want these models to do right now.

François Chollet
Grokking is a very, very old phenomenon. We've been observing it for decades. It's basically an instance of the minimum description length principle. Given a problem, you can just memorize a pointwise input-to-output mapping, which is completely overfit.

It does not generalize at all, but it solves the problem on the trained data. From there, you can actually keep pruning it and making your mapping simpler and more compressed. At some point, it will start generalizing.

That's something called the minimum description length principle. It's this idea that the program that will generalize best is the shortest. It doesn't mean that you're doing anything other than memorization. You're doing memorization plus regularization.

Dwarkesh Patel
a.k.a. generalization?

François Chollet
Yeah, that absolutely leads to generalization.

Dwarkesh Patel
So you do that within one skill. The pattern you see here of meta-learning is that it's more efficient to store a program that can perform many skills rather than one skill. This is what we might call fluid intelligence. So as you get bigger and bigger in models, you would expect it to go up this hierarchy of generalization. It generalizes to a skill, then it generalizes across multiple skills.

François Chollet
That's correct. LLMs are not infinitely large. They have only a fixed number of parameters. They have to compress their knowledge as much as possible. In practice, LLMs are mostly storing reusable bits of programs like vector programs. Because they have this need for compression, every time they're learning a new program they're going to try to express it in terms of existing bits and pieces of programs that they've already learned before.

Dwarkesh Patel
Isn't this generalization?

François Chollet
Absolutely. Clearly LLMs have some degree of generalization. This is precisely why. It's because they have to compress.

Dwarkesh Patel
Why is that intrinsically limited? At some point it has to learn a higher level of generalization and a higher level, and then the highest level is the fluid intelligence.

François Chollet
It's intrinsically limited because the substrate of your model is a big parametric curve. All you can do with this is local generalization. If you want to go beyond this towards broader or even extreme generalization, you have to move to a different type of model. My paradigm of choice is discrete program search, program synthesis.

If you want to understand that, you can sort of compare and contrast it with deep learning. In deep learning your model is a differentiable parametric curve. In program synthesis, your model is a discrete graph of operators. You've got a set of logical operators, like a domain-specific language. You're picking instances of it. You're structuring that into a graph that's a program. That's actually very similar to a program you might write in Python or C++ and so on. We are doing machine learning here. We're trying to automatically learn these models.

In deep learning your learning engine is gradient descent. Gradient descent is very compute efficient because you have this very strong informative feedback signal about where the solution is. You can get to the solution very quickly, but it is very data inefficient. In order to make it work, you need a dense sampling of the operating space. You need a dense sampling of the data distribution. Then you're limited to only generalizing within that data distribution. The reason why you have this limitation is because your model is a curve.

Meanwhile, if you look at discrete program search, the learning engine is combinatorial search. You're just trying a bunch of programs until you find one that actually meets your spec. This process is extremely data efficient. You can learn a generalizable program from just one example, two examples. This is why it works so well on ARC, by the way. The big limitation is that it's extremely compute inefficient because you're running into combinatorial explosion, of course.

You can sort of see here how deep learning and discrete program search have very complementary strengths, and limitations as well. Every limitation of deep learning has a corresponding strength in program synthesis and inversely. The path forward is going to be to merge the two.

Here's another way you can think about it. These parametric curves trained with gradient descent are great fits for everything that's System 1-type thinking: pattern recognition, intuition, memorization, etc. Discrete program search is a great fit for Type 2 thinking: planning, reasoning. It's quickly figuring out a generalizable model that matches just one or two examples, like for an ARC puzzle for instance.

Humans are never doing pure System 1 or pure System 2. They're always mixing and matching both. Right now, we have all the tools for System 1. We have almost nothing for System 2. The way forward is to create a hybrid system.

The form it's going to take is mostly System 2. The outer structure is going to be a discrete program search system. You're going to fix the fundamental limitation of discrete program search, which is combinatorial explosion, with deep learning. You're going to leverage deep learning to guide and to provide intuition in program space, to guide the program search.

That's very similar to what you see when you're playing chess or when you're trying to prove a theorem, for instance. It's mostly a reasoning thing, but you start out with some intuition about the shape of the solution. That's very much something you can get via a deep learning model. Deep learning models are very much like intuition machines. They're pattern matching machines.

You start from this shape of the solution, and then you're going to do actual explicit discrete program search. But you're not going to do it via brute force. You're not going to try things randomly. You're actually going to ask another deep learning model for suggestions. It'll be like, "here's the most likely next step. Here's where in the graph you should be going." You can also use yet another deep learning model for feedback like "well, here's what I have so far. Is it looking good? Should I just backtrack and try something new?” Discrete program search is going to be the key but you want to make it dramatically better, orders of magnitude more efficient, by leveraging deep learning.

By the way, another thing that you can use deep learning for is of course things like common sense knowledge and knowledge in general. You're going to end up with this sort of system where you have this on-the-fly synthesis engine that can adapt to new situations.

The way it adapts is that it's going to fetch from a bank of patterns, modules that could be themselves curves, differentiable modules, and some others that could be algorithmic in nature. It's going to assemble them via this intuition-guided process. For every new situation you might be faced with, it's going to give you a generalizable model that was synthesized using very, very little data. Something like this would solve ARC.

Dwarkesh Patel
That's actually a really interesting prompt. There’s an interesting crux here. I talk to my friends who are extremely optimistic about LLMs and expect AGI within the next couple of years. In some sense, they also agree that scaling is not all you need but that the rest of the progress is undergirded and enabled by scaling. You still need to add the System 2 and the test time compute on top of these models.

Their perspective is that it's relatively straightforward to do that because you have this library of representations that you built up from pre-training. It's almost like it's just skimming through textbooks. You need some more deliberate way in which it engages with the material it learns. In-context learning is extremely sample efficient. To actually distill that into the weights, you need the model to talk through the things it sees and then add it back to the weights.

As far as the System 2 goes, they talk about adding some kind of RL setup so that it is encouraged to proceed on the reasoning traces that end up being correct. They think this is relatively straightforward stuff that will be added within the next couple of years.

François Chollet
That's an empirical question so we’ll see.

Dwarkesh Patel
I assume your intuition is not that. I'm curious why.

François Chollet
My intuition is that this whole System 2 architecture is the hard part. It’s the very hard and unobvious part. Scaling up the interpolative memory is the easy part. It's literally just a big

curve. All you need is more data. It's an interpolative representation of a data set. That's the easy part.

The hard part is the architecture of intelligence. Memory and intelligence are separate components. We have the memory. We don't have the intelligence yet. I agree with you that having the memory is actually very useful. If you just had the intelligence but it was not hooked up to an extensive memory, it would not be that useful because it would not have enough material to work from.

Dwarkesh Patel
Former guest Trenton Bricken advanced an alternative hypothesis that intelligence is just hierarchically associated memory. When Sherlock Holmes goes into a crime scene he's extremely sample efficient. He can just look at a few clues and figure out who was the murderer. He's able to do that because he has learned higher level associations. It's memory in some fundamental sense.

Here's one way to ask the question. In the brain, supposedly we do program synthesis, but it is just synapses connected to each other. Physically, it's got to be that you just query the right circuit, right?

François Chollet
You are, yeah. It's a matter of degree.

Dwarkesh Patel
Training in the environment that human ancestors were trained in means you learn those circuits. If you train on the same kinds of outputs that humans produce — which to replicate, requires these kinds of circuits — wouldn't that train the same thing that is whatever humans have?

François Chollet
It's a matter of degree. If you have a system that has a memory and is only capable of doing local generalization from that, it's not going to be very adaptable. To be really general, you need the memory plus the ability to search to quite some depth to achieve broader and even extreme generalization.

One of my favorite psychologists is Jean Piaget, the founder of developmental psychology. He had a very good quote about intelligence. He said, "intelligence is what you use when you don't know what to do." As a human living your life, in most situations you already know what to do because you've been in this situation before. You already have the answer.

You're only going to need to use intelligence when you're faced with novelty, with something you didn't expect. It’s something that you weren't prepared for, either by your own life experience or your evolutionary history. This day that you're living right now is different in some important ways from every day you've lived before. It's also different from any day ever lived by any of your ancestors. You're still capable of being functional. How is that possible?

Dwarkesh Patel
I'm not denying that generalization is extremely important and the basis for intelligence. That's not the crux. The crux is how much of that is happening in the models.

Okay, let me ask a separate question about the differences in intelligence between humans. Maybe because of the reasons you mentioned, the intelligence tests are not measuring it well. But clearly there's differences in intelligence between different humans.

What is your explanation for what's going on there? That's sort of compatible with my story. There's a spectrum of generality and these models are climbing up to a human level. Even some humans haven't even climbed up to the Einstein level or the François level.

François Chollet
That's a great question. There is extensive evidence that differences in intelligence are mostly genetic in nature. That means that if you take someone who is not very intelligent, there is no amount of training data you can expose that person to that would make them become Einstein. This points to the fact that you really need a better architecture. You need a better algorithm. More training data is not in fact all you need.

Dwarkesh Patel
I think I agree with that. I might phrase it in this way. The people who are smarter have, in ML language, better initializations. If you look at the neural wiring, it's more efficient. Maybe they have greater density of firing.

Some part of the story is scaling. There is some correlation between brain size and intelligence. Within the context of "scaling" LLMs, people talk about architectural improvements. A model like Gemini 1.5 Flash performs as well as GPT-4 did when GPT-4 was released a year ago, but is 57 times cheaper on output. Part of the scaling story is that we're in like extremely low-hanging fruit territory when it comes to those architectural improvements.

Dwarkesh Patel
We're back now with the co-founder of Zapier, Mike Knoop. You're funding this prize and you're running this prize with François. Tell me about how this came together. What prompted you guys to launch this prize?

Mike Knoop
I've been AI curious for 13 years. I co-founded Zapier and I've been running it for the last 13 years.

I first got introduced to your work during COVID. I went down the rabbit hole. I had a lot of free time. It was right after you'd published your paper, "On the Measure of Intelligence". You introduced the concept of AGI and that this efficiency of skill acquisition is the right definition, and the ARC puzzles.

I don't think the first Kaggle contest had been done yet. It was still running. It was interesting but I just parked the idea. I had bigger fish to fry at Zapier. We were in the middle of this big turnaround of trying to get to our second product.

It was January 2022 when the chain-of-thought paper came out. That really awoke me to the progress. I even gave a whole presentation to Zapier on the GPT-3 paper. I felt like I had priced in everything that LLMs could do. That paper was really shocking to me in terms of all these latent capabilities that LLMs have that I didn't expect they had.

I actually gave up my exec team role. I was running half the company at that point. I went back to being an individual contributor and just doing AI research alongside Bryan, my co-founder. Ultimately, that led me back towards ARC. I was looking into it again. I had expected to see this saturation effect that MMLU and GMS8K have.

When I looked at the scores and the progress over the last four years, I was really shocked to see that we'd made very little objective progress towards it. It felt like a really important eval. As I spent the last year quizzing people about it in my network and community, very few people even knew it existed. If it's right that this is a really globally, singularly unique AGI eval — and it's different from every other eval that exists that more narrowly measures AI skill — then more people should know about this thing.

I had my own ideas on how to beat ARC as well. I was working nights and weekends on that. I flew up to meet François earlier this year to quiz him and show him my ideas. Ultimately I asked him why more people didn't know about ARC? You should actually answer that. It's a really interesting question. Why don't you think more people know about ARC?

François Chollet
Benchmarks that gain traction in the research community are benchmarks that are already fairly tractable. The dynamic is that some research group is going to make some initial breakthrough and then this is going to catch the attention of everyone else. You're going to get follow-up papers with people trying to beat the first team and so on.

This has not really happened for ARC because ARC is actually very hard for existing AI techniques. ARC requires you to try new ideas. That's very much the point. The point is not that you should just be able to apply existing technology and solve ARC. The point is that existing technology has reached a plateau. If you want to go beyond that and start being able to tackle problems that you haven't memorized or seen before, you need to try new ideas.

ARC is not just meant to be this sort of measure of how close we are to AGI. It's also meant to be a source of inspiration. I want researchers to look at these puzzles and be like, "hey, it's really strange that these puzzles are so simple and most humans can just do them very quickly. Why is it so hard for existing AI systems? Why is it so hard for LLMs and so on?"

This is true for LLMs, but ARC was actually released before LLMs were really a thing. The only thing that made it special at the time was that it was designed to be resistant to memorization. The fact that it has survived LLMs so well, and GenAI in general, shows that it is actually resistant to memorization.

Mike Knoop
This is what nerd-sniped me. I went and took a bunch of the puzzles myself. I've shown it to all my friends and family too. They're all like, "oh yeah, this is super easy. Are you sure AI can't solve this?" That's the reaction and the same one for me as well. The more you dig in, you realize there's not just empirical evidence over the last four years that it's unbeaten, but there are theoretical concepts behind why. I completely agree at this point that new ideas are needed to beat ARC.

There’s a lot of current trends in the world that are actually working against that happening. We’re actually less likely to generate new ideas right now. One of the trends is the closing up of frontier research, right? The GPT-4 paper from OpenAI had no technical detail shared. The Gemini paper had no technical detail shared, like the longer context part of that work.

Yet that open innovation and progress and sharing is what got us to transformers in the first place. That's what got us to LLMs in the first place. So it's actually a little bit disappointing that so much frontier work has gone closed. It's really making a bet that these individual labs are going to be the ones to have the breakthrough and not the ecosystem. The internet and open source has shown that it's the most powerful innovation ecosystem that's ever existed, probably in the entire world.

François Chollet
It's actually really sad that frontier research is no longer being published. If you look back four years ago, everything was just openly shared. All of the state-of-the-art results were published. This is no longer the case.

OpenAI single-handedly changed the game. OpenAI basically set back progress towards AGI by quite a few years, probably like 5-10 years. That's for two reasons. One is that they caused this complete closing down of frontier research publishing.

But they also triggered this initial burst of hype around LLMs. Now LLMs have sucked the oxygen out of the room. Everyone is just doing LLMs. I see LLMs as more of an off-ramp on the path to AGI actually. All these new resources are actually going to LLMs instead of everything else they could be going to.

If you look further into the past to like 2015 or 2016, there were like a thousand times fewer people doing AI back then. Yet the rate of progress was higher because people were exploring more directions. The world felt more open-ended. You could just go and try. You could have a cool idea of a launch, try it, and get some interesting results. There was this energy. Now everyone is very much doing some variation of the same thing.

The big labs also tried their hand on ARC, but because they got bad results they didn't publish anything. People only publish positive results.

Dwarkesh Patel
I wonder how much effort people have put into trying to prompt or scaffold, do some Devin-type approach, into getting the frontier models to produce good solutions on ARC. I mean the frontier models of today, not just a year ago. A lot of post-training has gone into making them better. There's Claude 3 Opus or GPT-4o.

I hope that one of the things this episode does is get people to try out this open competition. They have to put in an open source model to compete, but we could also figure out if maybe the capability is latent in Claude and just see if you can show that. That would be super interesting.

Dwarkesh Patel
Let's talk about the prize. How much do you win if you solve it? Let's say you get whatever percent on ARC. How much do you get if you get the best submission but don't crack it?

Mike Knoop
We have a little over a million dollars in the prize pool. We're running the contest on an annual basis. We're starting today through the middle of November. The goal is to get 85%. That's the lower bound of the human average that you guys talked about earlier. There's a $500,000 prize for the first team that can get to the 85% benchmark.

We don't expect that to happen this year. One of the early statisticians at Zapier gave me this line that has always stuck with me: "the longer it takes, the longer it takes." My prior is that ARC is going to take years to solve. We're also going to break down and do a progress prize this year.

There's a $100,000 progress prize which we will pay out to the top scores. $50,000 is going to go to the top objective scores this year on the Kaggle leaderboard. We're hosting it on Kaggle. We're then going to have a $50,000 pot set for the best paper that explains conceptually the scores that they were able to achieve.

One of the interesting things is we're also going to be requiring that in order to win the prize money, you put the solution or your paper out into the public domain. Typically with contests, you see a lot of closed-up sharing. People are private and secret. They want to hold their alpha to themselves during the contest period.

Because we expect it's going to be multiple years, we want an interactive game here. The plan is that at the end of November we will award the $100,000 prize money to the top progress prize. We'll use the down time between December through February to share out all the knowledge from the top scores and the approaches folks were taking. That way we'll re-baseline the community up to whatever the state of the art is and then run the contest again next year. We'll keep doing that on a yearly basis until we get 85%.

Dwarkesh Patel
I'll give people some context on why I think this prize is very interesting. I was having conversations with my friends who are very much believers in models as they exist today. First of all, it was intriguing to me that they didn't know about ARC. These are experienced ML researchers.

This happened a couple nights ago. We went to dinner and I showed them an example problem. They said, "of course, an LLM would be able to solve something like this." We took a screenshot of it. We just put it into our ChatGPT app. It didn't get the pattern.

So it's very interesting. It is a notable fact. I was playing devil's advocate against you on these kinds of questions but this is a very intriguing fact. This prize is extremely interesting because we're going to learn something fascinating one way or another.

With regards to the 85%, separate from this prize, I'd be very curious if somebody could replicate that result. Obviously in psychology and other kinds of fields, which this result seems to be analogous to, when you run tests on some small sample of people they're often hard to replicate.

I'd be very curious to know, if you try to replicate this, how does the average human perform on ARC? I'm also curious about the difficulty of how long it will take to crack this benchmark. It's very interesting thinking of the other benchmarks that are now fully saturated, like MMLU and MATH. Dan Hendrycks and Collin Burns who did MMLU and MATH, they were grad students or college students when they made it.

The goal when they made it just a couple of years ago was that it would be a test of AGI. Of course they got totally saturated. I know you'll argue that these are tests of memorization. But there's been a pattern we've seen. In fact, Epoch AI has a very interesting graph where you see this almost exponential curve. It gets 5%, 10%, 30%, 40% as you increase the compute across models, and then it just shoots up.

In the GPT-4 technical report, they had this interesting graph of the HumanEval problem set, which was 22 coding problems. They had to graph it on the mean log pass curve. Early on in training, or even with smaller models, they can have the right idea of how to solve this problem.

It takes a lot of reliability to make sure they stay on track to solve the whole problem. You really want to upweight the signal where they get it right at least some of the time, maybe 1/100 or 1/1000. They go from 1/1000 to 1/100 and 1/10 and then they just totally saturate it.

Here's the question this is all leading up to. Why won't the same thing happen with ARC? People had to try really hard with bigger models. Now they figured out these techniques like the ones Jack Cole has figured out that can get 35% with only a 240 million parameter language model.

Shouldn't we see the same pattern we saw across all these other benchmarks? You just eke out and then once you get the general idea, you just go all the way to a hundred?

François Chollet
That's an empirical question. We'll see in practice what happens. What Jack Cole is doing is actually very unique. It's not just pre-training an LLM and then prompting it. He's actually trying to do active inference.

Mike Knoop
He's doing test-time, right? He's doing test-time fine-tuning.

François Chollet
Exactly, he's doing test-time fine-tuning. This is actually trying to lift one of the key limitations of LLMs. At inference time, they cannot learn anything new. They cannot adapt on the fly to what they're seeing. He's actually trying to learn.

What he's doing is effectively a form of program synthesis. LLMs contain a lot of useful building blocks, programming building blocks. By fine-tuning it on the task at test time, you are trying to assemble these building blocks into the right pattern that matches the task. This is exactly what program synthesis is about.

I would contrast this approach with discrete program search. In discrete program search, you're trying to assemble a program from a set of primitives. You have very few primitives. For instance, people working on discrete program search on ARC tend to work with DSLs that have 100 to 200 primitive programs. It's a very small DSL but they're trying to combine these primitives into very complex programs. There's a very deep depth of search.

On the other hand, is what Jack Cole is doing with LLMs. He's got this vector program database DSL of millions of building blocks in the LLM. They're mined by pre-training the LLM, not just on a ton of programming problems, but also on millions of generated ARC-like tasks. You have an extraordinarily large DSL and the fine-tuning is very shallow recombination of these primitives.

Discrete program search is very deep recombination with a very small set of primitive programs. The LLM approach is the same but on the complete opposite end of that spectrum. You scale up the memorization by a massive factor and you're doing very shallow search. They are the same thing, just different ends of the spectrum.

I think where you're going to get the most value for your compute cycles is somewhere in between. You want to leverage memorization to build up a richer, more useful bank of primitive programs. You don't want them to be hard-coded like what we saw for the typical RTS. You want them to be learned from examples. You also want to do some degree of deep search. As long as you're only doing very shallow search, you are limited to local generalization. If you want to generalize further and more broadly, depth of search is going to be critical.

Dwarkesh Patel
I might argue that the reason that he had to rely so heavily on the synthetic data was because he used a 240 million parameter model. The Kaggle competition at the time required him to use a P100 GPU which has like a tenth or something of the flops of an H100.

For context for the listeners, the frontier models today are literally a thousand times bigger than that. For your competition, submissions can't make any API calls, can't go online, and have to run on NVIDIA Tesla P100. It's significantly less powerful.

Mike Knoop
There's basically a 12 hour runtime limit. There's a forcing function of efficiency in the eval.

François Chollet
But here's the thing, you only have 100 test tasks. The amount of computing available for each task is actually quite a bit, especially if you contrast that with the simplicity of each task.

Dwarkesh Patel
Basically, it would be 7 minutes per task. People who have tried to do these estimates of how many flops does a human brain have. You can take them with a grain of salt but as a sort of anchor, it's basically the amount of flops an H100 has.

Maybe you would argue that a human brain can solve this question in faster than 7.2 minutes. Even with a tenth of the compute, you should be able to do it in seven minutes. Obviously we have less than petabytes of fast access memory in the brain and these 29 GB or whatever in the H100.

Dwarkesh Patel
The broader point is that I wish there were a way to also test this prize with some sort of scaffolding on the biggest models, as a way to test whether scaling is the path to solving ARC.

François Chollet
Absolutely. In the context of the competition, we want to see how much progress we can do with limited resources. But you're entirely right that it's a super interesting open question, what could the biggest model out there actually do on ARC?

We actually also want to make available a private, one-off track where you can submit to us a VM. You can put on it any model you want. You can take one of the largest open source models out there, fine-tune it, do whatever you want, and just give us an image. Then we run it on the H100 for 24 hours or something. You see what you get.

Mike Knoop
It's worth pointing out that there's two different test sets. There is a public test set that's in the public GitHub repository that anyone can use to train. You can put in an open API call, whatever you'd like to do. Then there's the private test set, which is the hundred that is actually measuring the state of the art.

It is pretty open-ended and interesting to have folks at least attempt to use the public test set and go try it. Now there is an asterisk on any score that's reported on against the public test set because it is public. It could have leaked into the training data somewhere.

François Chollet
This is actually what people are already doing. You can already try to prompt one of the best models, like the latest Gemini or the latest GPT-4, with tasks from the public evaluation set. Again, the problem is that these tasks are available as JSON files on GitHub. These models are also trained on GitHub. So they're actually trained on these tasks.

That kind of creates uncertainty. If they can actually solve some of the tasks, is that because they memorized the answer or not? Maybe you would be better off trying to create your own private, ARC-like very novel test set. Don't make the tasks difficult. Don't make them complex. Make them very obvious for humans, but make sure to make them original as much as possible. Make them unique, different, and see how well your GPT-4 or GPT-5 does on them.

Dwarkesh Patel
There have been tests on whether these models are being overtrained on these benchmarks.

Scale recently did this with GSM8K. They basically replicated the benchmark, but with different questions. Some of the models actually were extremely overfit on the benchmark, like Mistral and so forth. Frontier models like Claude and GPT actually did as well on their novel benchmark as they did on the specific questions that were in the existing public benchmark.

I would be relatively optimistic about them just training on the JSON. I was joking with Mike that you should allow API access but keep an even more private validation set of these ARC questions. So you allow API access and people can play with GPT-4 scaffolding to enter into this contest. Maybe later on you run the validation set on the API. If it performs worse than the test set that you originally allowed the API to access, that means that OpenAI is training on your API calls. You go public with this and show them like, "oh my god, they've leaked your data."

Mike Knoop
We do want to evolve the ARC dataset. That is a goal that we want to do. François mentioned that it's not perfect.

François Chollet
Yeah, ARC is not a perfect benchmark. I made it over four years ago, almost five now. This was in a time before LLMs. We've actually learned a lot since about what potential flaws there might be. There is some redundancy in the set of tasks, which is of course against the goals of the benchmark. Every task is supposed to be unique in practice. That's not quite true. Every task is also supposed to be very novel, but in practice, they might not be. They might be structurally similar to something that you might find online somewhere.

So we want to keep iterating and release an ARC 2.0 version later this year. When we do that, we're going to want to make the old private test set available. Maybe we won't be releasing it publicly, but what we could do is just create a test server where you can query, get a task, and submit a solution. Of course you can use whatever frontier model you want there.

Because you actually have to query this API, you're making sure that no one is going to accidentally train on this data. It's unlike the current public ARC data, which is literally on GitHub. There's actually no question about whether the models are trained on it. They are because they train on GitHub.

By gating access to requiring this API, we would avoid this issue. For people who want to try whatever technique they have in mind, using whatever resources they want, that would be a way for them to get an answer.

Dwarkesh Patel
I wonder what might happen. I'm not sure. One answer is that they come up with a whole new algorithm for AI with some explicit program synthesis. Now we're on a new track. Another is that they did something hacky with the existing models in a way that actually is valid, which reveals that maybe intelligence is more of getting things to the right part of the distribution. Then it can reason.

In that world, that will be interesting. Maybe that'll indicate that you had to do something hacky with current models. As they get better you won't have to do something hacky. I'm also going to be very curious to see if these multimodal models will natively perform much better at ARC-like tests.

Mike Knoop
If ARC survives three months from here, we'll up the prize. We're about to make a really important moment of contact with reality by blowing up the prize, putting a much bigger prize pool against it. We're going to learn really quickly if there's a lot of low-hanging fruit ideas.

Again, I think new ideas are needed. Anyone listening might have the idea in their head. I'd encourage everyone to give it a try. As time goes on, that adds strength to the argument that we've stalled out in progress and that new ideas are necessary to beat ARC.

François Chollet
Yeah, that's the point of having a money prize. You attract more people and you get them to try to solve it. If there's an easy way to hack the benchmark, that reveals that the benchmark is flawed. You're going to know about it. In fact, that was the point of the original Kaggle

competition for ARC back in 2020. I was running this competition because I had released this dataset and I wanted to know if it was hackable, if you could cheat.

There was a small money prize at the time. It was like $20K. This was right around the same time as GPT-3 was released. People of course tried GPT-3 on the public data. It scored zero. What the first contest taught us is that there is no obvious shortcut. Now there's more money. There's going to be more people looking into it. We're going to find out. We're going to see if the benchmark is going to survive.

Let’s say we end up with a solution that is not like trying to brute force the space of possible ARC tasks. It’s just trained on core knowledge. I don't think it's necessarily going to be in and of itself AGI, but it's probably going to be a huge milestone on the way to AGI. What it represents is the ability to synthesize a problem-solving program from just two or three examples. That alone is a new way to program.

It's an entirely new paradigm for software development. You can start programming potentially quite complex programs that will generalize very well. Instead of programming them by coming up with the shape of the program in your mind and then typing it up, you're actually just showing the computer what output you want. You let the computer figure it out. That's what is extremely powerful.

Dwarkesh Patel
I want to riff a little bit on what kinds of solutions might be possible here, and which you would consider defeating the purpose of ARC vs. which are valid.

Here's one I'll mention. My friends Ryan and Buck stayed up last night because I told them about this. They were like, "oh, of course LLMs can solve this."

Mike Knoop
Good. Thank you for spreading the word.

Dwarkesh Patel
They were trying to prompt Claude Opus on this and they say they got 25% on the public ARC test. What they did was have other examples of some of the ARC tests and in context explain the reasoning of why you went from one output to another output and now you have the current problem. I think there was also expressing the JSON in a way that is more amenable to the tokenizer.

Another thing was using the code interpreter. Do you think the code interpreter, which keeps getting better as these models get smarter, is just the program synthesis right there? What they were able to do was get the actual output of the cells, that JSON output, through the code interpreter, like “write the Python program that gets the right output here.”

Do you think that the program synthesis kind of research you're talking about will just look like using the code interpreter in large language models?

François Chollet
I think whatever solution we see that will score well is probably going to need to leverage some aspects from deep learning models and LLMs in particular. We've shown already that LLMs can do quite well. That's basically the Jack Cole approach.

We've also shown that pure discrete program search from a small DSL does very well. Before Jack Cole, this was the state of the art. In fact, it's still extremely close to the state of the art and there's no deep learning involved at all in these models.

We have two approaches that have basically no overlap, that are doing quite well. They're very much at two opposite ends of one spectrum. On one end, you have these extremely large banks of millions of vector programs, but very shallow recombination, simplistic recombination. On the other end, you have very simplistic DSLs, 100-200 primitives, but very deep, very sophisticated program search.

The solution is going to be somewhere in between. The people who are going to be winning the ARC competition and making the most progress towards near-term AGI are going to be those that manage to merge the deep learning paradigm and the discrete program search paradigm into one elegant way.

You asked what would be legitimate and what would be cheating. If you want to add a code interpreter to the system, I think that's great. That's legitimate. The part that would be cheating is trying to anticipate what might be in the test, like brute force the space of possible tasks and train a memorization system on that. You rely on the fact that you're generating so many tasks, millions and millions. Inevitably there's going to be some overlap between what you're generating and what's in the test set.

That's defeating the purpose of the benchmark because then you can just solve it with that and you need to adapt just by fetching a memorized solution. Hopefully ARC will resist that, but no benchmark is perfect. Maybe there's a way to hack it. We're going to get an answer very soon.

Dwarkesh Patel
Although some amount of fine tuning is valid because they have to use open source language models to compete here and they're natively language. They'd need to be able to think in the ARC-type way.

François Chollet
Yes. You want to input core knowledge, ARC-like core knowledge, into the model but surely you don't need tens of millions of tasks to do this. Core knowledge is extremely basic.

Dwarkesh Patel
If you look at some of these ARC-type questions, I actually do think they rely a little bit on things I have seen throughout my life. For example, something bounces off a wall and comes back and you see that pattern. I've played arcade games and I've seen Pong or something.

For example, you see the Flynn effect and people's intelligence, as measured on Raven's progressive matrices, increasing on these kinds of questions. It's probably a similar story where since childhood now, we actually see these sorts of patterns in TV and whatever, these spatial patterns.

So I don't think this is core knowledge. This is actually also part of the "fine-tuning" that humans have as they grow up, seeing different kinds of spatial patterns and trying to pattern match to them.

François Chollet
I would definitely file that under core knowledge. Core knowledge includes basic physics, for instance bouncing or trajectories. That would be included.

But yeah, you're entirely right. The reason why, as a human, you're able to quickly figure out the solution is because you have this set of building blocks, this set of patterns, in your mind that you can recombine.

Dwarkesh Patel
Is core knowledge required to attain intelligence? For any algorithm you have, does the core knowledge have to be, in some sense, hardcoded? Or can even the core knowledge be learned through intelligence?

François Chollet
Core knowledge can be learned. In the case of humans, some amount of core knowledge is something that you're born with. We're actually born with a small amount of knowledge about the world we're going to live in. We're not blank slates.

But most core knowledge is acquired through experience. The thing with core knowledge is that it's not going to be acquired in school for instance. It's actually acquired very early in the first 3-4 years of your life. By age four, you have all the core knowledge you're going to need as an adult.

Dwarkesh Patel
Interesting. On the prize itself, I'm super excited to see the open source versions, maybe with a Llama (70B) or something, and what people can score in the competition itself. I'm also excited to test specifically the scaling hypothesis and I'm very curious if you can prompt on the public version of ARC.

You won't be able to submit that to this competition itself but I'd be very curious to see if people can crack that and get ARC working there. Would that update your views on AGI?

Mike Knoop
It's really going to be motivating. We're going to keep running the contest until somebody puts a reproducible open source version in the public domain. Even if somebody privately beats the ARC eval, we're going to still keep the prize money until someone can reproduce it and put the public reproducible version out there.

François Chollet
Exactly. The goal is to accelerate progress towards AGI. A key part of that is that any meaningful bits of progress need to be shared, need to be public, so everyone can know about it and try to iterate on it. If there's no sharing, there's no progress.

Dwarkesh Patel
What I'm especially curious about is disaggregating the bets. Can we make an open version of this or is this just possible with scaling? We can test both of them based on the public and the private version.

Mike Knoop
We're making contact with reality as well with this. We're gonna learn a lot about what the actual limits of the compute are. If someone showed up and said, "hey, here's a closed source model and I'm getting +50% with it," that would probably update us. We'd think, "okay, perhaps we should increase the amount of compute that we give on the private test set in order to balance."

Some of the decisions initially are somewhat arbitrary in order to learn about what people want. What does progress look like? Both of us are committed to evolving it over time in order to be the best or the closest to perfect as we can get it

Dwarkesh Patel
Awesome. Where can people go to learn more about the prize and maybe try their hand at it?

Mike Knoop
Arcprize.org. It's live now.

Dwarkesh Patel
It goes live today. One million dollars is on the line, people.

Thank you guys for coming on the podcast. It's super fun to go through all the cruxes on intelligence and get a different perspective and also to announce a prize here. This is awesome.

Mike Knoop
Thank you for helping break the news.

François Chollet
Thank you for having us.