/mlst

Bold AI Predictions From Cohere Co-founder

March 24, 2026



Machine Learning Street Talk

Podcast Transcript

Revision: b81d2024

Topic headings include retention indicators: ● above average, ● average, ● below average (vs. similar YouTube videos)

Interested in licensing ReScript for your podcast? Get in touch with Tim Scarfe.

Contents

Overview 2

Cohere Origins and the Canadian AI Ecosystem 3

Google, the Innovators Dilemma, and Coheres Genesis 4

Killer GenAI Applications and Abstraction Levels 5

Building Software with Behavior and the Actor Model 7

Democratizing Software Development 8

AI Reliability, the Chinese Room, and Existential Risk 9

Compute Demand and Coheres RAG-First Differentiation 11

Enterprise Adoption, Privacy, and GPU Challenges 13

Product Building and Cloud Partnerships 14

Gaming, Strategy, and Organizational Lessons 16

Transformer Architecture and System-Level Optimization 19

Inference-Time Computation and the Nature of Reasoning 21

Hiring, Verifiers, and Advice for Young Developers 23

References 25

Overview

Ivan Zhang, co-founder of Cohere, sits down with Tim Scarfe for a wide-ranging conversation about building enterprise AI from the ground up. Ivan recounts dropping out of the University of Toronto to chase the startup bug, eventually co-founding Cohere with Aidan Gomez (a transformer paper co-author) and Nick Frosst after Google failed to commercialize its own invention — a textbook innovator's dilemma.

The conversation covers Cohere's RAG-first approach to enterprise AI, where models are trained to operate in air-gapped environments relying on external knowledge bases rather than memorized data. A healthcare case study stands out: a simple RAG feature cut doctor preparation time from 30 to 5 minutes. Ivan discusses the practical challenges of GPU allocation in Kubernetes, the tension between consultancy and product, and why the first version of any product will never be right without customer feedback.

A surprising detour into competitive gaming reveals Ivan's League of Legends and Elden Ring habits, and the organizational lessons they carry — communication discipline, not overloading the comms, and situational execution over grand plans. The conversation lands on a striking analogy: debugging LLM context is like understanding human decisions given the information available. "We are all just reasoning engines processing information."

The final third covers transformer architecture persistence, system-level optimization over model-only scaling, inference-time computation as a paradigm shift, and the challenge of capturing human thought processes (not just final answers) in training data. Ivan closes with advice for young developers: pair up with AI tools and build things fast. This is the best time to be a developer.

Disclaimer: This episode is part of the Cohere partnership series.

Cohere Origins and the Canadian AI Ecosystem

Tim Scarfe 00:00:00

How do you see supporting the startup ecosystem in Canada? I think Cohere plays[1]

Ivan Zhang 00:00:05

obviously, it plays a pretty significant role. I think we're 1 of the bigger players in the scene. We have to show that it's possible to build a really, really ambitious project here. I think we are like an existence proof that you don't have to be, you know, sort of in the valley to build something really, really ambitious. And not only do we have to be the existence proof, we have to, you know, actually put in the time and effort to do what's necessary for the scene. Right? Like, not just sponsoring it, but also providing our space as a as a place for people to gather and make sort of these interactions and connections sort of serendipity to happen. What do you think could be a killer Gen AI application in in a couple of years? Something that we're not even, you know, really thinking about now.

CNBC Narrator 00:00:50

This year's Disruptor 50 list is chock full of companies making breakthroughs in AI. Cohere develops natural language processing models that clients can deploy to build chatbots, search engines, and more. Cohere currently has strategic partnerships with Google and AWS. I think the most exciting thing for me,

Ivan Zhang 00:01:06

with this technology is that it sort of evens the playing fields between everyone. So just because I spent a few years of my life memorizing how to build back end systems in Go and my my mom who hasn't had the same, you know, amount of training in this area, us sitting in front of a computer, we should be able to accomplish the same thing. Hey, I'm Ivan. I'm 1 of the cofounders of Cohere, and I'm happy to be having this conversation. So I'm a proud Chinese immigrant. I came to Canada about 20 years ago. Went to computer science at the University of Toronto. That's where I met Nick and Aidan. And I actually dropped out of school. So my friend started a company in undergrad, and she raised some funding and she decided, Hey, why don't you come work for me full time? And I thought it would be a sort of a more better learning experience just working on the job compared to staying in school. And so just went for it, dropped out,

[1]Cohere — company

worked a few years at startups just building sort of back end systems. Eventually, the you know, I caught the startup bug, and that's when I messaged Aiden. Was like, Like, we should try try to start something together. And and at that point, we were already a few years into doing for AI, which is this independent research group that now we brought on to Cohere. And so, obviously, the space was going to be AI. Like, we thought there's something sort of there. You know, he he did the transformer, and so he thought, oh, why don't I just scale scope this architecture? You know, at the time, didn't really have the ambition to do anything with it, except, you know, obviously improved search. We saw it as in a moment where, hey, now computers can actually communicate with us and also understand us. What interesting software could we build now that silicon has this capability? So it seemed obvious to sort of take the leap and[2][3]

Google, the Innovators Dilemma, and Coheres Genesis

Ivan Zhang 00:03:10

start coherent.

Tim Scarfe 00:03:11

Why didn't Google want to do it themselves? That that's that's a very interesting observation that you made. Honestly, it's a textbook example, the[4]

Ivan Zhang 00:03:20

innovator's dilemma. Right? Like, you have this very, very profitable, whatever, like 90% margin business, which was search. It's really hard to deprioritize anything from that to actually invest in something completely net new, which was this new interface or this behavior that computers have. That's not the case now. They're very focused on this space. But at the time in 2019, yeah, there was like no interest whatsoever.

Tim Scarfe 00:03:56

So how has your role kind of grown, you know, since the beginning? Because obviously, you've been there right since the very beginning and now cohere is this huge global enterprise. How has it changed? Oh, man. I mean, I I've worn so many hats while I'm here, and and that's sort of what I expected.

[2]Ivan Zhang — person
[3]Attention Is All You Need — paper
[4]The Innovators Dilemma — book

Ivan Zhang 00:04:10

Know, at startups I worked at before, was a very sort of normal thing. Like, you know, when you work at a startup, you're expected to just roll up your sleeves and just do sort of whatever it takes to make the business succeed. And so when I started, I was literally just setting up Kubernetes clusters to crawl data to train the first version of our model. And then at some point, I managed the product team. At some point, I managed the business operations team. Honestly, I think the role of the founder is to fill these gaps where the business needs you until you can hire operational leaders to take that on, and then you move on to the next like, impactful thing. And so more recently, I, you know, started this this sort of synthetic data team, and now I'm moving sort of back to, like, this engineering role. How do you see Cohere supporting the startup ecosystem in Canada? I think Cohere plays and, obviously, it plays a pretty significant role. I think we're 1 of the bigger players in the scene, and Shopify being sort of the other, you know, big prime around here, I think we have to show that it's possible to build a really, really ambitious project here. I think we are like an existence proof that you don't have to be, you know, sort of in the valley to build something really, really ambitious. And not only do we have to be the existence proof, we have to, you know, actually put in the time and effort to do what's necessary for the scene, right? Like not just sponsoring it, but also providing our space as a as a place for people to gather and make sort of these interactions and connections sort of serendipitous,

Killer GenAI Applications and Abstraction Levels

Tim Scarfe 00:06:04

center serendipity to happen. So I I think we we play a big role, but I think we're not the only player. Yeah. That serendipity thing's really interesting because, you know, I think I think a lot of us kind of downplay the role of serendipity in in building startups. But that makes you kind of think, oh, is it quite a random process? But how do you handle setbacks? You know, like when you've tried something and it doesn't work.

Ivan Zhang 00:06:27

I mean, setbacks in general, you know, you just sort of take it on the chin and you learn what you can from it. It's also, you know, it's tough to sort of identify what you had control over and what you didn't have control over. And obviously the thing to learn is, you know, what did you have control over? And sort of say, How could I do this next time when I have the same sort of control? So yeah. I mean,

that that's just more general advice, I guess, on how I look at setbacks.

Tim Scarfe 00:07:00

Very interesting. So what do you think could be a killer Gen AI application in in a couple of years? Something that we're not even, you know, really thinking about now. I think it'll be a product or a feature that's sort of built into a product

Ivan Zhang 00:07:16

where you don't even notice that it's AI. Like, I I I kinda see Gen AI as, like, a component of an overall system, and I think beautiful products sort of hide the components away from the user. Like, don't have to think about it at all. And so interacting with it should be very natural. It should be either, you know, like you're interacting with a friend and you don't even think that sort of this AI thing under the cover, or it's sort of an existing workflow that is now much, much easier, and you barely notice that it's like, you know, a reasoning engine under the covers making your life easier. Yeah. I I really agree with that. I think since the beginning of the deep learning revolution, we're seeing the level of abstraction increasing.

Tim Scarfe 00:08:03

So at the very beginning, you needed to know about, you know, CUDA and compilers and hardware, and then it was PyTorch so software engineers could could get involved. And and now the abstraction is increasing even more to kind of, you know, applications and and agents and so on. And do you think there might be a kind of breakthrough level of abstraction where we really build completely different applications?

Ivan Zhang 00:08:23

Totally. Totally. I I think I think that that's sort of why we focused on embeddings very early on. I I thought it was very, very important to sort of create a data type for the computer to just, like, map a language in sort of these discrete processes. And I think the interfaces that people will build, you know, giving silicon this capability to just totally understand you end to end. Yeah. Think that that's going to unlock so many new interactions

Tim Scarfe 00:09:00

that people will have. Yeah. It's it's so cool because now just using language, we can generate code. We can build these systems. But I find increasingly, I'm building kind of, you know, agent based systems using the actor model. And the systems are just they're they're of a different character than the kind of software we we used to build before. And and kind of that's okay. But it it but I don't know. I guess it's a it's a different way of thinking about software that that enterprises are are going to slowly adopt. Yeah.[5]

Building Software with Behavior and the Actor Model

Tim Scarfe 00:09:30

And I think the way I've seen people build

Ivan Zhang 00:09:32

software companies has changed as well. You know, I met this founder the other day, and he was actually building up a repository of chat logs with AI assistants. So his team has to actually share sort of all their, you know, chat histories working through technical problems, and there's just now a repository of like, oh, how did you solve this before? Oh, I had this conversation with, you know, this assistant. And that's sort of now like a shared, like, repository of knowledge

Tim Scarfe 00:10:06

that, you know, his humans can actually look at and, you know, do the job better. Yeah. And it's so fascinating that we're building with behavior rather than instruction, if if that makes sense. I guess it's still both. We're we're instructing the the the language models to do things. But usually, it's different components interacting with each other, and then we iteratively refine. So we say, Well, let's see how this thing behaves. I like that. I don't like that. So I'm gonna refine the behavior, and I'm gonna rinse and repeat. And then eventually, we converge on a system that does the thing we want it to do. And that and that's just really, really fascinating.

[5] Actor Model — concept

Ivan Zhang 00:10:36

I don't think it's that much different before AI. Right? I think the iterative feedback process we used to address feedback by literally writing the code to change how the system behaves. Maybe this feature doesn't work quite right, we'll just change it. With AI, I mean, it's the same sort of process. It's just it's much easier because now the computer is totally steerable with language. So rather than, you know, writing Python or JavaScript, you're sort of just describing what you want, and, yeah, the agent will sort of either behave differently or help you change the code to change your system.

Tim Scarfe 00:11:24

Yeah. I mean, it's it's like, I've I've got this software that processes all the MLST transcripts, and now I can get it to do crazy stuff. I can say, every time Ivan makes a mouth smack, I want you to put it in the transcript. And every time he does this, I want you to put it in the transcript. And it's all just generated, and then I might detect a situation where you you you made a sound or there was a distraction which I didn't specify before. So I just added in, and it's just this really rich kind of process. And the amazing thing is, like, you just can't re I mean, how would I have written some software to do audio event detection before? It it just boggles the mind. Whereas now, I'm I'm I'm just kind of saying what I want it to do, and I'm just iteratively building it. I think the most exciting thing for me,

Democratizing Software Development

Ivan Zhang 00:12:03

with this technology is that it sort of evens the playing field between everyone. So, you know, just because I spent a few years of my life memorizing how to build back end systems in Go, and my my mom, who hasn't had the same, you know, amount of training in this area, you know, us sitting in front of a computer, we should be able to accomplish the same thing. Right? If I wanted to write really, really good distributed systems, I think, you know, my mom should be able to do the same thing in front of the computer, right? It's the same sort of piece of silicon. It has the same access to information, you know, on the internet or, you know, scraped on common crawl or something like that. Why can't we both produce software or videos or photos that

Tim Scarfe 00:12:58

the computer could literally do if there's a different human sitting behind it? Do you think that people hold language models up to an unreasonable standard? And what I mean by that is people say, oh, they're only sort of like 96% reliable, and that's not good enough yet. But as you just said, in a way, it's it's no different traditional to traditional software. So, know, when is software ever been reliable? And also, we're we're used to kind of reasoning about information on Twitter and in social spaces, and everyone has different opinions, we're kind of doing inferences, and we're we're finding our own path in life. And how is that so different? Yeah. It's not different. I I think it's

AI Reliability, the Chinese Room, and Existential Risk

Ivan Zhang 00:13:32

you know, I see it really as another tool for creation, and I think unfortunately it's sort of been marketed as this conscious thing that will take our jobs or, you know, create bioweapons to make humanity extinct. But I think if you approach it from the lens of like, look, this is another tool to help me create things in front of a computer. You know, think that human in the loop aspect is important to reinforce because at the end of the day, it is about helping us sort of do our work better. So I think it totally should be viewed with a lens of, hey, you know, this is an intelligent thing that needs our steer to actually

Tim Scarfe 00:14:26

be useful. And what what is your intuition about it not being an existential risk? I mean, for me, I I love John Sowell. And he made a very, very simple argument, which is that, you know, agency and intelligence is a physical phenomenon. So he's not saying we couldn't recreate it in principle, but you know, it the only existence proof is is in the brain. And there is a difference between simulating thing and the actual thing. So you simulate fire. It might have all of the behavior and, you know, dynamics of fire, but it doesn't burn. It you know, that there's a property which is missing. So for that reason, you can simulate intelligence until the cow comes home, but it's not gonna have the actual physical phenomenon that that that we have. I mean, you roughly agree with that? I don't know. I think I think this technology can go quite far.[6]

[6] Chinese Room Argument — concept

Ivan Zhang 00:15:08

I wouldn't say it's not possible to sort of recreate the human level intelligence. Personally I think there are many limitations today that can be solved. There are some net new essential risks. But like other technologies that came before it, like the internet, like cell phones, there's definitely ways that the internet has impacted society. It has changed how humans have behaved. So I think it's worth sort of discussing, you know, what, you know, the AI risks are, but that should not be the the only focus point. Right? Like, there's so much positive, productive value being created. It's I I think that's worth focusing our energy on rather than only on the risks. I think we should discuss some of the risks,

Compute Demand and Coheres RAG-First Differentiation

Tim Scarfe 00:16:11

but not yeah. That should not be all of our energy. So there's been an exponential increase of demand for compute, you know, just just going back. And it was it would have been surprising 10 years ago if you said, look, we're just gonna need so much more compute in the future. People might not have believed you, but now it's it's absolutely obvious. But do you think that trend will continue? For my own interests, I hope, you know, the costs will go down.

Ivan Zhang 00:16:35

But I I I do think that the cat's sort of out of the bag. Like, the the humans will want more and more flops in sort of whatever that form takes. I think the yeah. I think it'll just be an exponential increase. Like, the models will get better, then humans will want more of it, then we need sort of the compute to actually power it. How is Cohere differentiated from the other, you know, kind of similar startups? So we focus on training our models to behave well in these air gapped private and secure environments. So relying less on its internal memory and more so on, hey, you have access to these 4 or 5 or 20 or 100 internal knowledge bases. How do you solve problems in that sort of setting? And so we actually just train our model to do that. So we have human trainers who essentially act as this model with only sort of these, you know, internal knowledge sources that enterprises would have. Think your HR systems, financial accounting systems. And we only teach the model to be good at this rag setting, where you you can retrieve information, you ground on it, and you you generate your answers sort of grounded on this external information that you haven't memorized.[7]

Tim Scarfe 00:18:07

So can you go into a bit more detail on on on rag then? Because that that seems to be 1 of the most powerful things that that you folks have really, really optimized on. Like, can you can you give me an example where it's really kind of lifted an application up? The most surprising thing that I saw was just this last week.

[7] Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks — paper

Ivan Zhang 00:18:23

We were talking to a team building software for doctors in hospitals. So the idea is they added this new feature. It's powered by Command R using this sort of Rag approach, where they would just sort of fetch the most recent patient records so the doctors can actually look up, Hey, I'm about to see this patient. Give some materials to prep with. So this tiny rag feature that they built that should produce the summary about the patient reduced their prep time from 30 minutes to 5. So I think that's a very real example of like hey, know, AI is actually improving patient care and in a way where it's sort of built into an existing workflow. The doctor isn't sort of learning this net new piece of software. It's like okay, he's just doing exactly what he's about to do. It's just now that there's this little box that shows a summary, and it, you know, completely changed, like, how much time he can you know, he should invest, you know, doing research on a single patient.[8]

Tim Scarfe 00:19:37

What's

Ivan Zhang 00:19:38

Cohere's approach to privacy and security? So the nice thing about our approach is we we send you the weights, right? Like our customers are just running our containers in sort of air gapped environments where there's no there's no way for us to actually look at what you're using the model for. So all the usage data, you know, you you keep that. And obviously, have, you know, sort of APIs and and, you know, different ways to access the model, but the the sort of end state is, hey, we're running on premise. We're running in these, you know, private networks. Our model is just another piece of software, you know, like a database that you manage and and you can control the access for behind your firewalls. I mean, that that's a very important part of everything we do. It's it's sort of, hey. Like, we can't just talk back to the Internet. Like, we have to behave well in sort of these lockdown environments.

[8] Cohere RAG Documentation — documentation

Enterprise Adoption, Privacy, and GPU Challenges

Tim Scarfe 00:20:39

Do you give your customers advice? Alright. Because, you know, this is still quite a it's a complex technology. And even though they can control it, you know, inside their their enterprise, I'd you know, that that there are complexities that weren't there before.

Ivan Zhang 00:20:54

How how do you help them with that? Yeah. I mean, we we provide a lot of advice. I think a lot of people are excited about Gen AI, and they're not quite sure how to apply it in their organization. And so sometimes they even make too rash of a decision to try to give their users, whoever they may be, maybe internal employees or their actual external customers, they might be giving their users something they don't even want. And then they're just shipping AI, the users don't know what to do with it. And I think the most general advice I've given so far is just, hey, work backwards from the human. Right? Like, what is the human doing today? And sort of see where it makes sense to add, like, an automated reasoning process grounded on internal knowledge. And I think a lot of times it's sort of, yeah, it's just the human in the loop use case that makes the most impact. And then once you sort of get into your organization in a way that lets people, for the lack of a better word, play with it, then the ideas start flowing and people start seeing use cases here and there. And it really takes a domain expert to figure out where, you know, this automated reasoning thing should be applied. So, yeah, I think it it's been interesting seeing how adoption has happened in the in the past year. Yeah. Because I think,

Tim Scarfe 00:22:28

1 of the issues with GPU compute in the enterprise is is efficiency. So strangely, the cloud providers, right from the beginning, they virtualized the CPU stuff, but not the GPU stuff. And people just created GPUs everywhere in random subscriptions all all over the place. And clearly now, we've got a problem. Right? And, you know, these language models, they take up a whole load of GPU memory, and it's a real optimization problem to figure out, like, you know, I need to have different versions, need to have large ones, I need to have small ones, I need to have a model serving, you know, architecture and so on. I mean, how how do you help folks with that? Yeah. I think,

Ivan Zhang 00:23:08

you know, if if you look under the covers of Kubernetes, it was not built with GPUs in mind. We spent a lot of time just trying to make even, like, fractional GPU allocations work. Like, I think it's a lot of pioneering engineering, unfortunately. But it's a lot of fun, of course, trying to solve these hard problems. When we help with our customers, it's, hey, we've seen this problem before. Here's a container that, you know, does the same thing that you're looking for. Or, you know, here's a manifest that you can run. Here's a Helm chart. I mean, that's that's the sort of thing with these sort of on prem deployments. Like, everyone has, you know, like a slightly different setup, so it's slightly more hands on. But we're we're sort of learning along the ways that, you know, the things that make sense to generalize. But, you know, we're certainly open to the fact that the market is still super young, super new, and so nobody really hasn't figured it out. So some, you know, some assembly involved

Tim Scarfe 00:24:15

to get things running.

Product Building and Cloud Partnerships

Tim Scarfe 00:24:16

Yeah. On on that note as well, because, you know, completely agree. I I think that because of the complexity, every installation, every architecture will have its nuances. But then there's a bit of a consultancy trap. Not not that consultants is bad because it's a great way to build amazing relationships with strategic customers. But how how do you balance kind of like having a real core offering that is as generalizable as possible versus doing some real targeted strategic,

Ivan Zhang 00:24:40

you know, applications? Building products is more nuanced than looking at it as, you know, a consultancy or or not. Like, I think, obviously you should not or obviously we're not building our business to become a services shop or a consultancy but the reality is the first time you build your product and you put it out into the world, it's very unlikely that that is the right manifestation without sort of being battle tested by user feedback. Right? Without feedback, you're not going to build, you know, the perfect feature set right off the bat. And so, you know, for your first few customers, you're just going to have to work really hard to, you know, change the product, customize product for them so that you know

they feel compelled to use you. And then from that experience, from that process, you can build sort of generalized features that apply to other customers. But I think it's naive to think that you can build a product like a v 0 and have it be the perfect feature set that'll apply for every customer. No. I think you have to work really hard, do the calls and installation many, many times, and just,

Tim Scarfe 00:26:07

yeah, like, serve serve your customers. Right? Can you tell me about your relationship with the cloud vendors? How how you how you're partnered with them and and also some of your partnerships with with some of your top customers? I mean, what what are people building on Cohere? So we partner quite deeply with our, you know, very, very public partners

Ivan Zhang 00:26:26

and even for the private ones. Because the manifestation of Gen AI and the enterprise is so unclear, there's many tracks that we explore. So 1 way we've done this is, you know, the first step is to actually just get the model into their on premise system so that their employees could actually, you know, use this thing to either build features to power their existing apps, build sort of new projects, or just even use it for their day to day work. And so think like large enterprises, their HR team is completely enamored and swamped with questions every day. And now with an assistant that's sort of connected to standard operating procedures and policies and stuff like that, could then be the sort of first draft to either answer a question or even as now as this tool to sort of search up sort of HR policies for the employees in a self serve way. Obviously, go deeper, you know, if the enterprise needs more than that, go deeper. So if there's some sort of behavior or some, you know, knowledge that's important to have in memory, we we help the enterprise sort of build that into their version of the model, and that's a deeper collaboration. Right? And obviously, every step along the way, we sort of help them think through, hey, is this actually useful for the humans that you're trying to serve? Because really, that that's, that's the most important part, right? Like, that's how ultimately our customers will get value out of this technology. And so, you know, for us, it it's really important to think about the humans that they're trying to serve. Absolutely. As a quick digression, didn't didn't you have a gaming background? I played a lot of video games growing up. Yes. But I was I was never like a pro gamer or something. But but do do you still game today? Yes. Yes. It's it's 1 of the only ways to sort of get my mind off of work and just take like like a tiny sliver of a break every day. Yeah. Absolutely. I've I've got a a Meta Quest Pro,

Gaming, Strategy, and Organizational Lessons

Tim Scarfe 00:28:43

and it's so therapeutic. Like, even even just putting it on and just being in the menu screen, I just feel like the the stress melts away, and I'm in a I'm in a different world entirely. But what what are your kind of go to games? League of Legends is a big 1. That's been a staple my whole life. Lately, I've gotten more into,

Ivan Zhang 00:29:00

like, Dark Souls type games, like Elden Ring or, like, Wukong. Yeah. Something about just, you know, trying things trying, like, a really hard thing over and over again and, you know, getting obliterated, like, you know, a 100 times to, like, a really hard boss. Like, there's some peace I find in that. Is there any crossover

Tim Scarfe 00:29:26

between what you do in the gaming world and real life? Absolutely. I think it's it's taught me to

Ivan Zhang 00:29:32

just have high endurance staring at a screen for many, many hours. I think it's it's taught me, you know, the importance of teamwork, the importance of communication as well and not over communicating. I think at times, you know, you you hear the straights like, hey. You should, like, you know, communicate more with your teammates. But sometimes if you communicate too much, it's actually noise. Right? It's not helpful information for whoever's on the other side hearing it to make a decision. And so, yeah, like, not overloading the comms is sort of a thing in gaming where, you know, if if your teammate's going through a stressful, you know, 1 versus 5 situation, like, you don't want to overload them with information so that they could perform well. I think that very much so applies in the real world at work. It's not particularly helpful to info dump every piece of information. You have to you know, be a bit more intelligent in sort of what you share and when you share it. Yeah. I used to be a big Quake Live player.

Tim Scarfe 00:30:43

And with strategy, what I noticed was 2 interesting things. Like, we think of it in terms of top down. So we we try and create these global rules for everything. But what I noticed when experts were talking about the strategy was how situational it was. So it's like in this situation, you should do this. And in this situation, you should do this. And sometimes it felt incoherent, which is which is kind of weird. So there's always this kind of tension of how can I cohere the top down with the bottom up strategy?

Ivan Zhang 00:31:12

Yeah. You can have a grand plan of how, you know, the scenario is gonna play out or how the game is going to play out. But really, it's sort of in the execution and in the micro details, you know, in the in the situation that really determine the outcome. That is absolutely applicable to the sort of real world workplace. Yeah. I mean, could have a strategy for the quarter, but if the execution is not precise or you're sort of not you're not reacting to new information, yeah, things will not go well. So I think that's something I've I've

Tim Scarfe 00:31:50

I've learned in gaming. Yeah. Yeah. Because I I wonder how how you manage that in your organization because there must be an analogy in the sense that, you know, a lot of people in an organization at different points can't understand each other. Right? And and it's because in their situation, you know, this is their model, and it might seem incoherent or even nonsensical when they kind of explain their view to some to someone else, and there are loads of different ways of overcoming this. But how do you think about that? I actually think about it exactly like how think about debugging

Ivan Zhang 00:32:20

language models. When the language model isn't behaving the way I expect, I just think about, okay, what did it have in its context? Right? Did the search system retrieve the right documents? You know, so is the is the model actually reading the right information? You know, what what is the model trained on? You know, how has it been steered? What sort of feedback is it sort of reacting with? And so, you know, given all this information, I I put myself in their shoes and sort of think, ah, okay. Like, perhaps this is why they made that decision. And then that helps me sort of inform, you know, what kind of feedback to give next. Right? So that it's actually relevant to,

Tim Scarfe 00:33:02

you know, what their decision making process was given the context they had. I love that. It's almost like an, you know, a non intentional stance. Because I'm I'm the same. When I use a language model, you don't think that the language model has some grand agenda. You you know, you you just kind of say, oh, well, you know, it's my fault. I must have just put the wrong information in. And people do, you know, generally make reasonable decisions with the information that they have available to them at the time. So I think that's that's a great way of thinking about it. Yeah. I mean, we're all just reasoning engines

Ivan Zhang 00:33:31

processing information. Right? Yes.

Tim Scarfe 00:33:33

Yes. But we have this weird kind of thing. I don't know. Like, you know, some people say that language models have deception and intentions and motivations and so on. We seem to have that. But but do we? I don't know. Yeah. Bit of a galaxy brain thought there. Yeah. I think I think it's

Ivan Zhang 00:33:53

it's something in the context. Right? It's something in the context that makes us behave that way. Like, why does someone lie? Right? You should if you think through, like, okay, maybe they went through this experience in their life. They, you know, had this sort of reaction when they, you know, did the thing, and so that's sort of motivating, hey, like in sort of their context window, you know, they've distilled those experiences into, hey, I should lie in this situation. And so that's what they've been prompted with. That's what they'll do. Okay. Let's talk a little bit about technology.

Transformer Architecture and System-Level Optimization ●

Tim Scarfe 00:34:28

So I guess the the first question is, you know, there have been some really big landmark innovations over the the course of the years, you know, like LSTMs and transformers and so on. And, you know, so so transformers were were invented with with huge help from Cohere, of course, because, you know, many of you guys were were on the on the paper. We're still using transformers.

Ivan Zhang 00:34:51

Why is that? I think the simple answer is that it works. I think it's been proven time and time again in the market, in the economy, that it's sort of the most yeah. I mean, it's the best architecture. It's the most efficient 1. At the end of the day, like, we just want to run gradient gradient descent super efficiently over, you know, some distribution of data. And I think transformers are right now the best architecture to do that. They're the best way to search for the right circuits to learn

Tim Scarfe 00:35:29

the dataset of the world. So there's been, like, some innovation in terms of building on top of transformers. So, you know, there's things like, you know, RLHF, for example, or even process supervision for doing reasoning and things like model distillation and, you know, quite a few things like that. What what do you think about, you know, the lineage of of things that we're building on the top? And and do you think that will just continue?[9]

Ivan Zhang 00:35:51

Yes. I think that will continue. I think, you know, the things you mentioned are sort of like 4 of hundreds of little tricks, you know, being applied to these models to make them, you know, commercially viable. It's going to be, you know, these tiny tricks that bring 10% to 15% improvements here and there, and it all compounds with each other. And, yeah, I think there's there's much more to be learned from just optimizing these models.

[9] Lets Verify Step by Step — paper

Tim Scarfe 00:36:22

And can you give us any hints on potential surprises in model architecture in the next couple of years? Do do you have any any insights on that? I think the models

Ivan Zhang 00:36:32

will become leaner. I think we'll be much better at determining, hey, for this set of capabilities, we only need models at this scale. And so, you know, perhaps a small model is good enough for these things. I think generally as a field, as we industrialize the products, we'll stop looking at the models as being, you know, the only thing we sell, and we'll start looking at them as being, you know, just sort of 1 component of an overall system.

Tim Scarfe 00:37:11

And so maybe more generally, the perception of the model will be slightly less than sort of the system that that we're building around it. Yeah. That really resonates with me because I I guess at at the moment, there's this paradigm of a monolithic model, and we spend loads and loads of money upfront and, you know, curate all of the data and train it and tune it and so on. And then we don't do anything to it for a few months until we make the next 1. And potentially in the future, it could be far more heterogeneous, for want of a better word. And the the process of learning just never really stops. And you never really know which particular model you're you're talking to because it just becomes this big system. I mean, is that is that am I completely off the mark there? I think it might look something like that. I think it might look something like,

Ivan Zhang 00:37:55

you know, there's just more gains to be had tuning other parts of your the other components of your system than just sort of, hey. Like, you know, the reasoning engine will do its job. I think at the end of the day, know, the reasoning engines are, you know, quite good these days, and they'll they'll do its job. Like, the generative model will do its job. I think what people will notice is like, hey. These other parts of the system that the reasoning engine will relies on to actually produce a good answer, they need to be improved as well. Right? Like, the interface maybe of how we're receiving information from the users could be improved as well. How we retrieve that information, you know, could be improved. So, like, stuff like re ranking was a surprising component, right? Like we never thought about it. We thought, hey, you know, we throw things in a database and just have the model search over it, and that'll be good enough. But you know, re ranker is 1 of those things that's come out of this industrialization. Right? Like, it's like, hey. It's not good enough to just throw in throw embeddings at the model. So I think there'll be more stuff like that that will discover either new components or components that need to be improved in the system

Inference-Time Computation and the Nature of Reasoning ●

Tim Scarfe 00:39:12

that's not just focused on the reasoning engine. What are your thoughts on this potential new paradigm of inference time computation? Are are you a fan of that idea? I think it's really, really exciting.[10]

Ivan Zhang 00:39:24

I think it's really cool that, you know, all the flops aren't just spent on backpropagation. And I think it's it's yeah. It's an exciting new technique. We're excited to explore it more, obviously, and I think it makes a ton of sense. Yeah. That's all I'll say.

Tim Scarfe 00:39:45

Yeah. I'd I'd it's probably secret squirrel, so I can't I can't probe too much, but I I'm I'm so excited about it. I mean, would you would you mind giving me your kind of hot take definition of reasoning?

[10] Adaptive Inference-Time Compute — paper

Ivan Zhang 00:39:57

Objectively, it's given some information you have in the context, produced produce more. Right? I think produce more context that could be helpful for the next token. That that is sort of the most MLE way I look at it. Right? Like, it could be, you know, sequences of a chess board state. Right? Sequences of, you know, like a like a math problem. Like, I think the the the point is that the model should produce tokens, which are helpful for finding finding the solution. I think that that's ultimately what reasoning is. And I think, yeah, it'd be applied to, you know, sort of any any sequences.

Tim Scarfe 00:40:43

And where does the reasoning come from? So, I mean, obviously, there there's there's the base model. There's the RLHF model. And, you know, some folks are and I've I've Aiden told me that that you folks are doing this. So, you know, almost like generating synthetic datasets and training the model how to reason from first principles, which is really interesting. And and then I guess the model could continue to learn how to reason from the users of of your system, and it just creates this virtuous circle.

Ivan Zhang 00:41:09

But is is there any more you can speak to on that? I guess 1 thing I'd like to to with synthetic data, like, generally in the field, is, like, it's not just, you know, the model teaching itself a thing. It's mostly you know, the models are great at expanding data distributions. Right? Like distributions of data that you care about. You know, some sometimes the data is super hard to find. And so if but you want more of this distribution, you could actually use the model to create more. And so, you know, when we talk about synthetic data in the field, it's about you still need the sort of real data, but you're sort of steering the model to create more and more of it. That's definitely being employed in teaching these models reasoning. But really, I think the problem that the field is trying to solve for is when you sample from these models, you're sampling its thoughts. Really, you're giving it chances to you know, pick the right next token to answer the question. But when we collect these demonstrations from humans, we're not necessarily capturing the human's thoughts. You know, when in in the most naive case, maybe, like, 2 years ago, you would show a human's prompts that, you know, you're trying to teach the model, and then they would write down the answer. And so all you're teaching the model is, hey, for this prompt, like, here's the answer. But the the the the thing you're trying to teach the model actually is how to think like the human to arrive at the answer. So how how did you reason about this prompt to actually arrive at, yeah, like this perfect answer? That's sort of more difficult

problem to solve for, which is like sort of prompting the humans to actually write down their thoughts in a way that

Hiring, Verifiers, and Advice for Young Developers

Tim Scarfe 00:43:10

we can optimize over. Yeah. It it's so interesting. I mean, I interviewed Ryan Greenblatt, and he'd he'd got a really good result on the ARC challenge by doing some chain of thought prompting and then sampling about 30,000 trajectories. And and in many cases, 1 of those trajectories was the correct answer. And he had a a post hoc verifier, and he was kinda building this whole reasoning engine. And it's interesting that we can execute code as well. So, you know, the language model can think. It can say, here's a function that generates the nth digit of pi. I can now run the function on a Python interpreter. I can pass the results back in. So, like, when when we start to think about these things systemically, it's mind blowing what we can do. But there is always this kind of thing that at some point, the amount of compute is is just gonna be very, very large because we're gonna be doing so much work potentially

Ivan Zhang 00:43:54

at inference time. Yeah. I think verifiers are a very exciting way to sort of have the model explore a system, generate sort of experiences that it can learn from. Yeah. I'm I'm pretty excited about sort of where to spend compute that's not just, you know, creating descent over the Internet.

Tim Scarfe 00:44:16

Tell me a little bit about the you know, how how do you hire people in into your team? Like, what what what kind of, you know, folks do do do you hire?

Ivan Zhang 00:44:24

Cohere, we have a good mix. So I think most folks are from a traditional, you know, background with a PhD, or they're from other labs, or something like that, or they worked on something, you know, pretty, pretty impressive. I personally, really like folks from untraditional paths into AI just because I also took an untraditional path to AI. So I really look for folks who worked really, really, really hard on a very technically challenging thing. So I think that shows to me the ability to spend a lot of flops, human flops on a problem to to actually solve it. Right? I I think yeah. I think that that's sort of

Tim Scarfe 00:45:20

basically all I look for. Yeah. That that's really interesting. So is is that another way of saying autonomy as well? Because I think a lot of the challenge of autonomy is is overcoming uncertainty and finding relevant information and being able to make progress and maybe discount things that that didn't work. But when when when folks can do that on their own, you know, without, you know, with without help, that that's a really good signal. Yeah. Absolutely. I I I basically just try to think,

Ivan Zhang 00:45:47

like, can this person do it better than, you know, the AI assistants?

Tim Scarfe 00:45:53

Like learn on their own and look for information on their own. So the entire scene is is changing. Right? So imagine that that you're giving advice to someone who's about 17 or 18, about to go to university, excited about, you know, all of this new technology and everything that's gonna be happening. What what advice would you give to them? I think

Ivan Zhang 00:46:11

it's a very exciting time to completely embrace the technology. Right now, the highest output as a human that you could have is actually just pairing yourself up with these assistants to, you know, build whatever you want to build. So completely embracing sort of using these tools, adopting them, learning the quirks of, you know, these sort of code editors that are AI assisted. Yeah. I think it's like the best time right now to be a developer. Like, could learn anything super fast. You could build anything super fast. I I would just say, like, dive deep and don't don't hold yourself back by you know, it's good to be skeptical, but, like, don't, you know, keep an open mind. Right? Amazing. Well, Ivan, it's been such an honor to have you on on the show. Thank you so much for joining us. Yeah. Thanks for having me. I enjoyed this conversation.

References

[1] **Cohere**
https://cohere.com/
company

[2] **Ivan Zhang**
https://ivanzhang.ca/
person

[3] **Attention Is All You Need**
https://arxiv.org/abs/1706.03762
paper

[4] **The Innovators Dilemma**
https://www.amazon.com/Innovators-Dilemma-Technologies-Management-Innovation/dp/1633691780
book

[5] **Actor Model**
https://en.wikipedia.org/wiki/Actor_model
concept

[6] **Chinese Room Argument**
https://plato.stanford.edu/entries/chinese-room/
concept

[7] **Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks**
https://arxiv.org/abs/2005.11401
paper

[8] **Cohere RAG Documentation**
https://docs.cohere.com/v2/docs/retrieval-augmented-generation-rag
documentation

[9] **Lets Verify Step by Step**
https://arxiv.org/abs/2305.20050
paper

[10] **Adaptive Inference-Time Compute**
https://arxiv.org/abs/2410.02725
paper

[11] **Getting 50% SOTA on ARC-AGI with GPT-4o**
https://redwoodresearch.substack.com/p/getting-50-sota-on-arc-agi-with-gpt
paper