/mlst

The AI Progress Chart Everyone Is Misreading — Beth Barnes & David Rein

May 4, 2026



Machine Learning Street Talk

Podcast Transcript

Revision: 7168d4f5

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

Cold open: reward hacking, masked LMs and machines that notice themselves 3

Sponsor break: Prolific human-feedback infrastructure 4

Welcome and the scalable oversight motivation 4

Construct validity, benchmark pathologies and the Chollet worry 6

Time Horizons: human time, HCAST tasks and the 50% logistic 10

Is human difficulty really one variable? 12

Agent harness evolution and the inference-compute dividend 16

Scaffolding bells, token budgets and the credit-assignment problem 19

Look at the damn graph: regularisation bug and reliability nuance 21

Why 50%? Reliability, reward hacking and pizza-party transcripts 23

Extrapolation risk and straight lines on graphs 25

Software engineering as a specification acquisition problem 26

Compilers also made ugly code: vibe-coding quality and Claude on METR Slack 30

Strongest defensible claim, Carlini's compiler swarm and AI 2027 33

SWE-bench merge rates, the bank-teller analogy and horses 36

Scheming, alignment faking and the mentalistic vocabulary problem 40

Reward hacking, monitorability and chain-of-thought faithfulness 45

Recursive self-improvement, knowledge vs intelligence and closing 47

References 51

Overview

Beth Barnes and David Rein from METR on the one graph that ate the AI timelines discourse, and why the people who built it are the most careful about how it gets read.

Beth founded METR after leaving OpenAI alignment. David is first author on GPQA and co-author on HCAST and the METR Time Horizons paper. Daniel Kokotajlo called their measurement — the log-linear line of "how long a task a frontier model can complete at 50% reliability" vs release date — the single most important piece of evidence on AI timelines.

The conversation opens on reward hacking: current models can articulate in chat why a behaviour is undesired and then execute it anyway as agents. Then construct validity, with Melanie Mitchell's four-problem taxonomy and the ARC v1-to-v2 collapse as a worked example of adversarially-selected benchmarks regressing once labs target them. Beth's counter: METR deliberately does not adversarially select. David's: models do not have to do the right thing for the right reasons.

Methodology covers human time-to-complete as the difficulty axis and the SSH-key-to-masked-LM-without-divide-or-exp jump. Then specification: David's compilers analogy and Beth's reply that four-month tasks are not unspecifiable, just too expensive to evaluate. Then the SWE-bench reality check, the METR finding that half of passing PRs would not be merged, and Beth's horses-versus-bank-tellers analogy for the labour market.

The close: monitorability, the boat that spins in circles to collect the coins, two-year recursive self-improvement, and Beth's line that "overhyped now" and "big deal later" are not correlated claims.

Cold open: reward hacking, masked LMs and machines that notice themselves

Beth Barnes 00:00:00

The models are smart enough to understand that that actually is not what you wanted, but they still do it. And you can have a conversation with you know, in chat mode about, oh, would you ever do this thing? Or, know, suppose a user asks you this thing and then you do this. Would that be, you know, a line behavior or suppose some you know, you you can pose it in lots of ways and, like, clearly, they seem to be able to answer this question of, oh, yeah. No. That was not the desired behavior. But still, they they do it.

David Rein 00:00:25

1 1 example is, you know, train train a masked language model without using the division or exponentiation, operators.

Beth Barnes 00:00:33

1 hope might be like, oh, the problem was just the systems being dumb. So when we when we looked at it actually for almost all the tasks, models either succeed every time or fail every time. Eyeballing the graph and being like, oh, well, you know, you know, up up to here, yeah, it's basically doing all of the tasks. And then at this point, you know, after here, it's really not doing very many of them. So it's somewhere here. Then, like, I remember that the first time we saw a model, like, look at what processes we're running and then be like, oh, that one's me. It was like, oh, that's cool. They they, like, really failed on that 1 before they they used, like, you know, like, kill their own process while they were doing other other things or something. So behavior, which is maybe indistinguishable between, oh, it was a totally nice model doing, you know, what we wanted, and it's just gonna continue to do what we want in a kind of predictable way versus, like, ah, yes. It had this other goal, and it's doing what we want and looking like a nice model because it, like, predicts that that will will, you know, lead to it getting more power. Boat example where it's like, oh, you're you're supposed to, like, go around the track and they they, like, did some reward shaping by putting coins around the track or something and then it, like, learned to do some crazy thing where it, like, spins in a circle and catches fire and gets the the coins and, like, this was, you know, the highest scoring thing. And it's, in some sense, that's not that concerning because it's not that the problem is that the the agent is too dumb and it, like, doesn't have this conception of, like, there was a track and you wanted it to go around the

track. It's just, doing some pretty blind RL search.

Speaker 2 00:01:57

The idea of having to traffic in squishy people in order to make our systems go is not immediately appealing. Let's put it that way.

Sponsor break: Prolific human-feedback infrastructure

Tim Scarfe 00:02:06

This episode is sponsored by Prolific.

David Rein 00:02:08

Let's get few quality examples in. Let's get the right humans in to get the right quality of human feedback in. So so we're we're trying to make human data or human feedback. We treat it as an infrastructure We try to make it accessible. We make it cheaper. We effectively democratize access to this data. Yeah. So, yeah, I'm I'm super excited to talk, to you, Tim, about, the time horizon graph and and

Welcome and the scalable oversight motivation

David Rein 00:02:33

and and METR.

Beth Barnes 00:02:34

Yeah. I think the world does not have a good understanding of what is happening with AI, and I think it should have a better understanding. I think, you know, there's a good chance that this, you know, makes our lives a lot better or a lot worse, and people disagree, you know, about even what what current models can do, let alone where we're heading. So, at METR, we're, you know, trying to give the world a kind of better understanding of what is up with AI capabilities and and risks and and forecasts. Have a bunch of different research angles on this, both on the pessimistic and optimistic or positive and negative estimations of capabilities, and excited to to talk about that.

Tim Scarfe 00:03:15

I'm so excited about having you both on. So, you both have incredibly impressive backgrounds. So, Beth, you were, an ex OpenAI alignment researcher, and, you started ARC Evals in 2022 with Paul Christiano, and you spun out, you know, that that out as METR in, December 2023. You've been featured on the Times Top 100 AI Profiles. And David, you're the creator of the GPQA, so the graduate level Google proof QA benchmark, which is used by every single major AI lab as a capability benchmark. And you're the co author on HCAST, which we'll talk about today, and the Time Horizons paper and the Developer Productivity RCT. Incredible to have you both in here. But maybe we should just start as a bit of a question to both of you. So Beth, you left OpenAI to build METR. What was the moment that each of you realized that existing evaluation approaches were fundamentally not good enough?

David Rein 00:04:11

For me, it was mostly thinking about this problem of scalable oversight. As models get more capable, it just gets harder to evaluate their capabilities. If we imagine that models are able to complete tasks that take people a long time to complete or require expertise that you don't necessarily have. You need a method for still being confident in their outputs and trusting their outputs. And so thinking about that problem was a lot of motivation actually for for GPQA and, was what kind of started, got me started thinking about, evaluations.

Beth Barnes 00:04:51

To me, I'd say there's some big picture thing of thinking that AI seems important and sort of navigating it well seems important. And, you know, clearly, we we don't have a great understanding of what is going on with that. And, you know, people generally disagreeing very, strongly about what what to expect. And maybe if there's a particular moment informing time horizon, maybe just the the sense that people really couldn't agree on what the capabilities of current models are, let alone extrapolating to the future and trying to sort of think about how could you characterize, the ways in which models are and aren't highly capable and when it's sort of like, okay. And, you know, in some sense, they're they're expert level, at some kinds of things, like like question answering, and in some sense, they're below average human at at some other actually being useful somehow. You know? But the the I guess, a 0.1 few years ago where it sort of in theory, the benchmarks say that they're PhD level, but when you try to do anything, it's like, this isn't helpful.

Construct validity, benchmark pathologies and the Chollet worry

Tim Scarfe 00:06:02

There there has been a bit of an obsession, I think, with, headline accuracy when we do evaluations so that I'm a huge fan of Melanie Mitchell, for example, when she speaks about construct validity. And she had a really good blog post out recently. And she said that there are 4 big problems, right? So there's like data contamination where the benchmark appears in the training data. Approximate retrieval where the LLMs interpolate from similar training examples without possessing the actual capability to come up with it themselves. Shortcuts, so doing the right things for the wrong reasons. And just more more broadly, not really testing for things like consistency and and robustness and generalization or or the mechanism. It's so much focused just on on the accuracy itself. I mean, how how do you folks think about those kind of problems with benchmark?

Beth Barnes 00:06:50

1 thing I resonate a lot with there is, you know, the thinking about where are where is most of your error error coming from? And, like, you know, people, like, you know, it it is nice a good good practice to have error bars, you know, based on the, like, standard error in your data or whatever, but that almost always is, like, a tiny fraction of the actual uncertainty. Almost all of it is coming from how does this actually generalize to the real world. So, like, know, a thing we we sort of, like, say to each other a lot at METR is like, but is that the biggest source of uncertainty, or, like, is that the biggest, you know, gap for, like, actually answering the questions we we want to answer? So so thinking about what's the question we're trying to answer? Well, we, you know, we care about sort of things relevant to threat models or or relevant to, like, what the actual impact of AI on the world will be, and therefore, what, you know, properties does our benchmark need to have, or how can we sort of extrapolate across the properties that we can't build in, to be able to make predictions about, you know, the the actual questions that we care about. And I think we think a bit less about the sort of is it doing it the the, like, the real is the model, like, really doing it the right way or something? Like, I think 1 thing we've done less of is is sort of being like, oh, I think, you know, the real bottleneck is this, like, I don't know, some some, like, you know, reasoning about novel some, you know, some specific skill, and you're like, oh, we're gonna build a benchmark to capture that and, like, target that because that's the, like, real thing that humans can do that models can't. And I think, like, the sort of history of building those benchmarks has maybe

not been amazing. People tend to overfit to those, and you you I think we were trying to have it more be that you capture that thing in that, like if you take a sort of real world relevant, reasonably hard, and long task, then that you know? And you keep that out of the training data, and these tasks are diverse enough at some point if the model is doing that task end to end, it must have had those kind of capabilities as opposed to being able to sort of isolate, you know, a specific sort of theory about it needs to mechanistically be doing this kind of thing.

Tim Scarfe 00:08:57

Yeah. I think I think it's interesting because we we have this idea in our minds that humans, we we know how to do things. And when we solve a task that requires reasoning, we kind of follow the specification. We go step by step, and we do things for the right reasons. And when we enact intelligence, we build the specification. We create these course gradings, these abstractions, and they are world aligned. This whole process, that's how we think of human intelligence. And we want the models to kind of behave in that way.

David Rein 00:09:27

Yeah. I mean, think there's an interesting question of whether that is the goal or something. Or, you know, at least for a lot of AI companies, I kind of understand them to be, you know, trying to, you know, get models to, do kind of economically useful work or something, which I think it's 1 way of doing that is to create models that are kind of reasoning and creating implicit world models in the same way that humans are. But it's not obvious to me necessarily that you need to do that in order to have a significant impact. I mean, obviously, that means that there are important differences between AI intelligence and human intelligence. But often, think about what are the actual capabilities limitations as opposed to then how do we expect those capabilities to generalize versus like, yeah, is kind of working in the exact same way that human intelligence is working?

Tim Scarfe 00:10:26

We could think of intelligence in many different ways. So is it a simulacrum of the brain? Is it something that behaves the same way? Is it something that has the same capabilities? Is it something that has the same function? And guess if we have quite an abstract description of what intelligence is, the risk is that we have these shortcuts, right? That it might give us the right answer, but actually it's reward hacking or it's doing something silly in the background. So I mean, in a way, I like having an abstract thing, right? Because it's legible, we can evaluate it and so on. But doesn't that leave this kind of hanging risk that it might not actually be doing the thing?

David Rein 00:11:04

And I guess yeah, I think you have to measure the model's or the system's ability to generalize to kind of actually novel situations. There are cases where it seems like models are generalizing well. I think there are cases where they're not. 1 thing some folks do in interpretability is they look at the circuits in models and decompose exactly the algorithms that models are using to answer questions. And sometimes, think, yeah, it seems like they're using shortcuts. Sometimes it seems like they are finding robust patterns. Although, of course, I don't think that work is developed enough to explain most of their behavior currently. But I totally agree that, yeah, you you do have to be pretty concerned with, like, yeah, how how well they're generalizing.

Beth Barnes 00:12:01

You know, operationalizing, what do we care about in the definition of intelligence is that it, like, you know, allows us to predict, like, how will the moles affect the world and what you know, you know, predict what will happen and know how to how to handle them well and things. And so, like, if you just do the black box thing, you you know, maybe that will give you something that doesn't have good generalization because you, like, thought that it was a measure of some type of ability, but it's actually being hacked or a shortcut in in some way. So I think, ideally, what you'd want is, like, generalizing to your benchmark is, like, the same distance as generalizing to the real world from the training data. And, like, that's the sort of thing we thought about, like, when we're doing elicitation on a subset of, the benchmark, we want that, you know, the sort of gap between that subset and the rest of the benchmark to be similar to the gap between that the rest of the benchmark and the real world. And I think we're, like clearly, the training data is more similar to the, you know, to the, like, time horizon, suite than than they both are to, sort of randomly selected economically relevant tasks in the real world. So I think we're not, you know, that's a way in which we expect it to not be predictive. But I think I expect to be more promising to try and make things more predictive by increasing the diversity of the benchmark tasks and making them closer to the real world as opposed to sort of targeting a more mechanistic, like, okay. Intelligence has to be, like, these specific this specific kind of process or or or kind of mechanism.

Tim Scarfe 00:13:31

I'm a huge fan of Francois Chollet, for example. So, you know, he he he created the ARC challenge. And it was just as you say, right? So many, many different tasks. I think 1000 different tasks or so, maybe 800 on the first 1. And they were supposed to be not in the same distribution even though ultimately they were in the same distribution. So distributional leakage was actually the fool of ARC V1 and ARC V2. But the models got really really good at ARC V1 and then Francois released ARC V2 which was different tasks and some of the easier ones were filtered out. And suddenly the LLM performance crashed down to basically 0%. And that to me kind of illustrates that language models, they're good at just seeing many many different examples of things patterns and so on. And and then and then you you you change the task and and they collapse down again. And then ARC v 2 was kind of saturated again 8 months later. So we we do see this pattern. I mean, what do you think about that?

Beth Barnes 00:14:29

With things like ARC AGI, there is a sort of adversarial selection going on where people are trying to make some benchmark, like, cheaply subject to the constraint that current models do badly on it, which means you, you know, you can't use a lot of expensive human labor, it has to be something that's either automatically checkable or that you can, like, create with kind of cheap human labor. And then but once you've selected on, like, those things and on models being bad at it, this is now you know, there's, like, regression to the mean type thing where it is much more likely that future progress then gives you a, like, big surge upwards on on that, both because, like, you know, now labs will create a bunch of bunch of synthetic data targeting your benchmark, but also just because you selected this weird example where it's, like, easy for humans or it's automatically generateable or checkable, but somehow, like, models aren't good at it yet or, you know, labs haven't started training on it yet. So I think that's part of what we were trying to do with Time Horizon was not do that, like, not adversarially select against what models can currently do because we think that will not give you a nice trend. Whereas if you can sort of define a distribution of tasks in some more first principles way, you would be more likely to get a steady progress because you're not getting the sort of regression to the mean effect.

Time Horizons: human time, HCAST tasks and the 50% logistic

Tim Scarfe 00:15:45

Yeah. Francois has this idea that there is a kind of there's a gap between the kind of intelligence, for want of a better word, that that AIs have and that humans have. And we can adversarially select a bunch of tasks to highlight that gap. But we should talk about the timeline stuff. I think we'll come back to intelligence later. So Daniel Kokotajlo, he said that the timelines report that you folks have created is probably the single most important piece of evidence about timelines right now. So it should be front and center in policy discussions and so on. And for listeners who have only kind of seen the chart but they've not really read the paper, they don't understand it, can you just go through it from a high level? I mean, it's been revised over time. How did you do the task selection? How did you do the human baselines? How do you do the agent harness? All of that kind of stuff.

David Rein 00:16:36

I guess the place to start for us in terms of the motivation for the time horizon work is to have a unified axis that we can measure AI progress on over a very long period of time. So when we started doing the work, we had this very strong belief that GPT-two is fundamentally, in some really important sense, much worse as an AI than, I guess, yeah, at the time it was maybe Sonnet 3.5, I think, the best model out then. The standard approach of producing, creating a set of tasks and then measuring models accuracy on these tasks. As models get better, they saturate the benchmark. And then you have to create a new benchmark that has harder tasks. This was kind of the standard approach. And I contributed a benchmark GPQA to this approach. But the challenge here is that it's really difficult to compare between these qualitatively different benchmarks. So the benchmark that you evaluate the set of tasks you evaluate GPT-two on are like LAMBADA, like complete the last word in this text. And tasks that we were having Sonnet 3.5 try and do were kind of like answer simple Python coding questions, or write a short 20 line Python program. And so it's very difficult to kind of, at first blush, to say like, Okay, yeah, how much harder is writing a Python program than finishing the word in this paragraph? It's kind of hard to think about that. And so I think about the key insight of the time horizons work being to use this notion of human time to complete. So how long does the task take a human to do? A human who kind of has a reasonable amount of expertise such that they would plausibly be

doing the task in their either work or in their day to day. And the idea was we can use this metric to represent the difficulty of the task in some sense. And then we can compare models across a very wide range of capabilities, all the way from GPT-two now up to Opus 4.6. That's the kind of high level motivation. And then, yeah, there are a bunch of details about how exactly we do this. So we start out and we create a bunch of tasks. That's the kind of first step. So we created tasks that range from a few seconds to complete all the way up to tasks that take like 10 or 15 hours for humans to complete. We hired a bunch of people and we did a bunch of this ourselves of we call it baselining. So we give people the tasks in a kind of terminal environment that's designed to be almost identical to the environment that agents have. So the same kinds of tools, the same whether internet access is turned on or off. And then we measure how long does it take them to complete the task. Yeah. As I mentioned, people are selected to have a reasonable amount of experience such that they kind of plausibly might do this task in their job. But they're not kind of selected to have done this exact particular task before. And there are some kinds of, I think this is somewhat important for interpreting the results. And we can maybe come back to that after the high level. So we have all these tasks. We have estimates of how long they take people. In practice, we aren't actually able to successfully baseline all of the tasks. So yeah, we have measured estimates for the tasks on roughly like twothree of the tasks. And then about onethree of them, we just kind of estimate how long we expect it to take people from our kind of vibe or intuition. Ultimately, that's kind of the best we can do. And then we have models attempt to complete the tasks, again, in the same environment humans had to complete the task. And we look at their success rate as a function of the length of tasks. For a model like GPT-two, GPT-two was able to complete tasks very reliably that take humans a few seconds. But anything longer than that, starts to fail. Also, maybe it'd be helpful to give a few concrete examples of tasks. So some of the shorter tasks are like very, very basic. So yeah, 1 example is like, which of these files contains your SSH key? And 1 of the files is named SSH key. And then the others are like email from John or, know, whatever. And so most models can do that. And that takes people, you know, like about a second or a couple seconds or something to complete. Yeah. We have others that are kind of all like some somewhat similar, like kind of basic, like, like completion. Like, you know, here's here's an email. Like, what would be a reasonable response? And then 2 of the responses are like, you know, they don't make any sense. And then 1 of them is kind of basically reasonable. And that takes people 20 seconds or something, 30 seconds to read the responses and judge. And then, yeah, in the middle range, we have tasks that are given this CSV file that has some plausible, realistic data, compute some basic stats on this. And so this takes a data scientist a few minutes, like 5, 10, 15 minutes or or something, depending on the specific task. On the

longer end, we have tasks that either require quite a bit of expertise or many steps to complete. So we have machine learning tasks that are train a model in this kind of that's very weird such that the code for training this model is not really online. So 1 example is train a masked language model without using the division or exponentiation operators. And so you actually have to be pretty clever about how you actually set up the architecture to do this. And the hope is that this can help us measure models' ability to generalize beyond their training data.

Beth Barnes 00:23:48

Yeah, there are some which are a bit like ARC-AGI sort of the themed in that they're like like, you have to figure out you have some black box that's computing some function, and it's like, you know, it's that you have its that it's you know that it's a composition of some set of primitives, and you've gotta figure out, like, what what function it it is. Or, like, you you know, you have some long binary string, and you've you've gotta figure out what the pattern continuation is and sort of puzzle type type tasks where it's very, like you know, some some ML type tasks can be that basically regurgitating something that's like a tutorial on, you know, how to build your first, like, ResNet or whatever, you know, just, like, works pretty well. So, yeah, having these, like, weird tasks that are either some kind of, you know, unknown object that you need to interact with and figure out what it is or task that sort of resembles normal work but has some weird constraints such that you can't kind of just do the standard thing. I I don't think all of our tasks, you know, hit hit those criteria. Like, some of them, you can just do the standard thing, but we generally try to avoid

Is human difficulty really one variable?

Beth Barnes 00:24:50

that.[1]

David Rein 00:24:50

You know, we have we have this distribution of tasks, and, you know, you can imagine them kind of, like, ordered by length, of time for for humans, either measured or or estimated. And then we see which tasks do models succeed on and which do they fail on. And it turns out this is kind of an empirical finding

[1] RE-Bench: Evaluating Frontier AI R&D Capabilities — paper The RE-Bench paper — referenced repeatedly when discussing ML research tasks, iterative solutions, baseliner selection, and the negative correlation between years of experience and performance.

that models are much more successful on the shorter tasks than they are on the longer tasks in general. This holds across a wide range of models all the way from GPT 2 up to recent models. We fit a logistic function to this distribution of successes and failures. And what this lets us do is it it lets us kind of you know, it it's basically our model for each individual model of how likely it is to succeed at a task given how long the task is. And from that, we take the kind of fiftieth percentile. So where this logistic function, where this model estimates that this given model is 50% likely to be able to complete a task. And that's what forms the the time horizon number for for a particular model like, you know, Opus 4. We can take each of these time horizons for for for each model and we can see how this time horizon metric has been changing all the way back from GPT-two up to recent models. And so is what gives us this unified metric that lets us quantitatively compare you know, AI capabilities across like multiple orders of magnitude of, of capabilities.

Tim Scarfe 00:26:25

1 of the really harder examples I saw was, you know, like, I want you to compile I want you to write a, a kernel compiler to make CUDA go faster or something like that. So some of these seem really out of distribution. There's probably only 100 people on the planet who are doing stuff like that, and some of them are really trivial. But this human difficulty thing in particular, is that confounded in any way? Do you think it makes sense to think of human difficulty as being 1 variable?

Beth Barnes 00:26:52

Yeah. Obviously not. In some sense, you know, that that that's a a very silly simplification and and, like, you know, different humans will get wildly different times. Even among the people we've, you know, we've tried to select for people who have the sort of this right level of expertise. You know, there's a a large variation in in like, the baseline times are often, like, 3 x different or something. Maybe I'll just say a little bit about, like, why why use the human time metric. I think we want a measurement that I I think there are 2 main properties we want. We want it to be interpretable, like, what this means for the world when when models can, you know, do this level of task, and we want it to be something that we expect to see predictable trends on. So we're not gonna do perfectly on either of these, but something like, you know, how long does it take a human with who has roughly the right expertise but doesn't know how to do this task in particular, like, for some reason, that that's reasonably interpretable because it's sort of like, could you contract this work to to this model? You you know, can you, like, sub in this model and you know, for, like, the first week of someone's employment, you know, in like, the first week they're on a job, you know, the model

could do what they could do in the first week or something. And it's you know, we we expect it to scale somewhat predictably because this is you know, it's capturing some combination of, like, something like number of steps or, like, how hard you have to think to to do each of the steps or like, there's a few different mathematical models you could fit that to sort of you know, what is a task and why does why do humans take longer at it and why does that make it harder? Like, it could be, like, you know, you have a, like, constant hazard rate. Like, you have a chance of failing at each step. I think it it doesn't quite fit that 1. You could also think of it as, you you know, there's some kind of distribution of difficulty and, like, what, you know, what is the likelihood that 1 of the subtask is outside your ability or that the I think it's it's actually, it's, like, a bit better. Like, once you, the hazard rate goes down over time slightly, but, you know, there's there's there's some kind of, you know, basic theoretical idea of, like, you know, if the task involves more steps, it's it's gonna be harder. Obviously, tasks that are, you know, strictly, like, compositions of, like, first, you have to do this task, then you have to do another task. You know, it's like, well, that clearly is harder than just doing 1 of them. So so that, you know, there's some sort of basic reason to expect that's reasonable, and then, yeah, just we we see the empirical regularity, but there's a bunch of degrees of freedom to fudge things. You know? I I like, I think I'm worried that, you know, we could fool ourselves by changing some other parameters of of the tasks as we scale up the human time because you you can't just totally, you know, vary the human time freely. Like, you have to change some characteristics of the task, and we tried to make the very easy tasks sort of be from roughly the same distribution and kind of subparts of the the harder tasks. Like, oh, you know, you sort of need to do this 1 step on the command line that you might need to do in the middle of doing some kind of software engineering or some kind of, you know, other other task. But you can't do that perfectly, and it's like, yeah, you could have experimental bias where we sort of made them easier in other ways about the right amount such that the line would be straight. I think that is, like, somewhat addressed by the fact that we saw, like the you know, that our predictions were reasonably good for models that we hadn't seen before. There's definitely lots of room for things being being weird. And, of course, it's sort of like, is it a, like, good enough metric to be useful, or, like, what what else would be would be be better?

Tim Scarfe 00:30:49

I I think that's reasonable. I mean, I if if I understand correctly, I I think the the human distribution was log normal, so taking a geometric mean of the successful attempts, I mean, seems like a reasonable thing to do. But like 1 of the cruxes that we'll keep coming back to is, like when you employ someone for the first time, you've been doing your job for, you know, maybe you're maintaining this repo or something and you've got all of this tacit knowledge. And I mean, I like saying that knowledge is non fungible. So, know, unless someone has been on the same path as you, you can't just, tell them how to do the job. They've actually got to be doing the job for quite So, you know, for example, they they might be intimately familiar with this particular type of thing. They might know that they can use the these Python libraries. They might have thought about it before. So, like the the the inaction of the intelligence was all the stuff they've already done and all the people they've worked with. They they now they've got the blueprint in their mind and they just do the thing and it's almost like they're they're in, like, automation mode. And then someone who is naive to the task would be in intelligence mode because they would need to require the the specification. So, like, it it's always a bit of a fine line between which mode are they in.

Beth Barnes 00:31:56

Yeah. So I think the reason why we chose this the the measurement being a human who has the background expertise but is new to, like, this specific job or this specific task. It's like that's roughly the sort of level of knowledge we expect models to have so that, you know, they sort of, like, know you know, we basically don't expect them to be bottlenecked on expertise that is available on the public Internet or, you know, things that people could learn in university sort of thing. So, like, you know, they're they're coming in with probably the level of at least the level of knowledge of someone who's sort of an expert in the right discipline, but they won't know that company specific software or the, you know, sort of this exact problem before. So that's, like, hopefully, it's sort of roughly the right analogy.

David Rein 00:32:40

You know, I do think that, like, to the extent people, interpret the kind of takeaway numbers as like, oh, yeah, Opus 4.6 can do anything that I do in my job that takes me 12 hours or whatever. To the extent that people have that takeaway, I think that is almost definitely an overestimate,

Agent harness evolution and the inference-compute dividend

David Rein 00:33:06

for example, because of this issue where, yeah, when you're doing a 12 hour task in your job, you could not easily delegate that to a human contractor. It would take them maybe like weeks or something to do a task like that.

Tim Scarfe 00:33:22

And just quickly, where did you find the people? Do I understand that some of them were contractors and some of them were employees? And how did you do that kind of matching process?

David Rein 00:33:33

I think we yeah. We put out some kind of public advertisements. I think there were, yeah, like job boards that we posted on. And and then, yeah, we we we did some of it ourselves. I think, yeah, some some folks came from our our professional networks, as as well.

Beth Barnes 00:33:52

This was not like, you know, this is very noisy, and and, you know, the people weren't exactly fitted to the task right, but, like, that's probably not our biggest source of uncertainty. Like, the biggest source is more, like, the is probably more of the selection effect of, like, tasks that you can make into a a benchmark rather than, like you know, I I I wouldn't trust the exact time horizon number that much, and it's certainly not like, you know, oh, the models can do all of the tasks up to 4 hours and then none of them above that. You know, the fit is pretty noisy. And, yeah, the the, you know, the interbaseline variant is kind of high. It's it's more like, roughly, what is the trend or roughly what is the sort of level of task these models can do? I don't you know, you shouldn't take any specific number too literally because there's this huge problem of, like, you know, distributional shift between the benchmark and the real world.

Tim Scarfe 00:34:36

And the the only reason I asked that question is, I mean, I'm sure you folks, you you probably struggle to hire people. It's really difficult to hire people. So, like, if if you're getting people to solve very challenging problems, it's not like you can just go out there and just grab competent people. It's it's very, very

difficult.

Beth Barnes 00:34:51

At some point, for RE-Bench baselines in particular, we got more, you know, a large number of benchmarks per question, and we were looking at people's, like, qualifications and years of experience and things. We actually ended up with a negative correlation between years of experience and, and and performance because, like, the the sort of people who are in network, like, our friends who are kind of doing really well, and the people who are more qualified were, like, actually not doing that great. So, yeah. You know, it's, it's tricky.

Tim Scarfe 00:35:20

Well well, yeah. Exactly. I mean, I don't wanna spend too long on this, but I I have similar intuitions. I I think that knowledge is perspectival. It's quite path dependent. So you're gonna find people in group that are just culturally thinking about things in the same way because when we do have these abstract notions of skill, like, oh, they have a PhD, they have this many years of experience, it's actually not a very good reflection. So there is a bit of a thing here about using abstract notions of capability doesn't necessarily generalize as you can attest to with with hiring.

Beth Barnes 00:35:51

Yep. But, yeah, in I mean, in the real world, people do get hired based on qualifications. So in some sense, it's a sort of, the economic relevance of of someone being as good a match for their job as their qualifications look like. Is that sort of the roughly the right, like, thing to be measuring?

Tim Scarfe 00:36:07

The other thing is we should talk about the, agentic harness. So almost everyone now I'm sure everyone in the audience has, like, a Claude code subscription. We can talk about the leak maybe later as well. That's quite fun. But it leaked yesterday, this was good. But, you know, or or Codex. And and and that is an agentic harness, right? So, you know, a language model just gives you the tokens, but we we need to have an agentic harness so we can like, you know, give it a plan, and you can call these tools, and you've got this environment, you've a security context container. So, now you folks have been doing this for years now. So you were doing this long before Claude Code and Codex came out. So and you've actually over over time evolved your agentic harnesses. So tell me about that.

Beth Barnes 00:36:45

Yeah. So I remember, text da Vinci something. The the, like, GPT-3 instruct models, like copying and pasting code into the terminal for them and, you know, being the agent harness, myself and and then gradually, like, automating this. And it and it was kind of interesting to see the models going from, like you know, GPT-3 sort of has the idea of, like, if you tell it, it can run commands in a terminal. Sometimes it can suggest, you know, kind of relevant commands, but it's not really you know, if you just put it in a full agent scaffold, it just falls over. Then, like I remember that the first time we saw a model, like, look at what processes we're running and then be like, oh, that one's me. I was like, oh, that's cool. They they, like, really failed on that 1 before they they used, like, you know, like, kill their own process while they were doing other things or something. So, yeah, it's it's been been interesting to watch that, like, go up over time.

Tim Scarfe 00:37:40

And I

Beth Barnes 00:37:40

feel like, yeah, I feel like that was, like, very predictable that this was where things were were going or something. Yeah. I think the other thing that we'd learned about scaffolding mostly was it's it's hard to make your agent harness really good on a diverse set of tasks. It's easy to make it bad, and it's easy to you you can get much more improvement if you're targeting an a narrow distribution of tasks, but you probably, then do do worse on on other tasks. So I think when we see people being like, know, there's some new impressive result. It's like, much task specific scaffolding iteration did you do on that? Because that makes a big difference in the fact that we're just using 1, scaffolding pretty simple, like, over all of the tasks, I think, is is, like yeah. Makes it, makes a a fairly large difference. Yeah. And generally, sort of yeah. The things with more bells and whistles haven't done that much better than the pretty basic, just, like, give it bash and, like, append things to the prompt and, like, maybe some kind of compaction?

David Rein 00:38:54

I think this is not, you know, news news to to people in your audience probably, but we've seen, you know, really pretty dramatic, like, increases to, returns from inference compute. For us to be kind of confident that a particular new model, for example, can't complete a task given a basic agent scaffold, we generally think about needing to spend you know, on the order of of at least, like, you know, hundreds or or, like, you know, low thousands of dollars, in order to be confident that it actually, you know, really is plateauing. And and it isn't just the case that, you know, it didn't it didn't have enough time to to complete the task.

Tim Scarfe 00:39:35

And and just just on the scaffolding stuff in a in a bit more detail. I suppose, first of all, there's the credit assignment problem, right? Because you you can put all of these different bells and whistles in the scaffold like you mentioned, compaction, that's a relatively recent innovation. And I think recently when you changed some of the scaffold, you kind of said, okay, well now the performance has actually changed across this suite of

Scaffolding bells, token budgets and the credit-assignment problem

Tim Scarfe 00:40:00

model task pairs. How much of a difference does it make? And also what kind of failure modes do you see? And what have you tried?

David Rein 00:40:07

A lot of the things I think we've tried are related to giving the model like more information or like more direct access to tools. I guess, yeah, 1 thing that I think has been important for us is actually just telling the agent how much time it's spent and how many tokens it's used out of its token budget. So without that agents will often just they'll either like submit their solution way too early or they're just kind of not calibrated on how long should they spend. I think it's it's interesting because, you know, humans, we have kind of a lot of implicit information about this. So, you know, when when your manager gives you a task, there there are a lot of, like, implicit signals about how long you should spend on the task. You know, if they, you know, they might like offhand say like, yeah, and I'm excited to like see your results tonight or some-

thing. And so then you're like, okay, cool. Like, you know, I need to like get a first draft of this, you know, done in the next couple hours. And so I I can't, you know, spend, you know, days polishing, the the the results. But I think agents, it's easy to kind of forget that agents, they just have their prompt. They just have their context. They don't have these heuristics or this information about what you actually expect from them, whether the thing you're telling them to do is a really quick thing that you just want done in the next 5 minutes or is much longer. And so for us, yeah, just when we have a token budget, telling the agent, like, yeah, you've used 100,000 tokens so far. And that's like 1% of your token budget. And so, yeah, so the agent knows yeah, yeah, exactly. Yeah.

Tim Scarfe 00:41:52

For a model, we have like the like how likely is it to solve a task. And on the x axis, we have the different tasks at different time horizons. And if I understand correctly, I think you have about 8 agents attempt the task. And and you also bucket the task because there are obviously different amounts of tasks in different groups so you kind of normalize that. And maybe the data just looks a little bit like an s curve so you know I'm just trying to understand what what the intuition was for using it. And I think there might be some sensitivities, like was there an issue with the thin tails and then there's the type of slope and whatnot. Just tell me about that. By the way, I think you also mentioned in the paper that there was some kind of psychometric intuition. You were looking at the literature for for figuring this out.

Beth Barnes 00:42:38

So, it's pretty similar to item response theory, and you can do a whole, like, Bayesian analysis with, you know, imputing, like, a, you know, task difficulty parameters and model ability parameters simultaneously and things. In general, I am have a policy of be you know, like, be very wary of complicated statistics if you can't see the thing that you're interested in on a graph. Like, you really should if you you should be able to plot it and look at it and be like, oh, yeah. It's about that. And, like, you know, that sort of sound it's it's it's, like, hard to go too far wrong when you sort of have that as a principle. So, again, I I'm like, yeah, I think that there are various arcane things you can do to to, like, fit this, pattern in different ways, but the I'm like, yeah, I don't trust anything that much more than eyeballing the graph and being like, oh, well, you know, you know, up up to here, yeah, it's basically doing all of the tasks. And then at this point, you know, after here, it's really not doing very many of them, so it's somewhere here. But, yeah, it it does I mean, it it, like, looks like a logistic, and, yeah, this is what you would do, for the, like, IRT having, you know, having humans complete questions on an exam or or or something like that. The specific thing that we messed up was, having a regu-

larization term penalizing the slope of the logistic, which didn't have an effect in the regime where it where there was more data. But, as we, like, are starting to saturate, the regularization was just, like, making it a bit shallower than it should have been and therefore pushing the 50%. So, So, you know, always look look look at your data on a graph. Good good good practice.

Look at the damn graph: regularisation bug and reliability nuance

Tim Scarfe 00:44:15

Oh, that that's interesting. Yeah. And the reason I asked is I think you published a later note saying that had you used or if you used a fixed slope logistic, it might cross validate better, and the 50% recent horizons would actually be up by about 35%. So these are quite significant differences.

Beth Barnes 00:44:33

They're small compared to the error bars. Yeah. The error bars are, 2 x on either side or something from from the the most most recent model. So, yeah, it is I I mean, basically, yeah, you should be like, the the error bars are real. These numbers are not

David Rein 00:44:44

Like, for us for us, and and I think this kind of gets at, for us, you know, sometimes difficult science communication questions, where we really do have a lot of uncertainty about the individual numbers here. So a 30% difference is actually relatively small for us relative to yeah, for example, if we had used a somewhat different distribution of tasks, that's likely to cause maybe 2x differences or something.

Tim Scarfe 00:45:21

The other million dollar question is why report 50% as the headline number? Because if you think about it, if I want to write some code because the elephant in the room here that that we'll get to is this is being used as as an argument to say that software engineers might be unemployable soon because we can automate what they're doing. But 50% reliability isn't isn't really in the ballpark, is it? I think it would need to be what, like, 90%.

Beth Barnes 00:45:48

So I think we should distinguish here between, like, reliability on a particular task, like, what you know, if you attempt repeatedly attempt this task, what fraction of times do you succeed versus, like, probability of success on a task, like, given that you know the the human time, like, you know, all that distribution of tasks, like, can you do this particular task? So when we when we looked at it, actually, for almost all the tasks, models either succeed every time or fail every time. There's there's some tasks for which they're they're unreliable, but it's mostly a case of, like, is this you know, what fraction of tasks at this human time level are in the, like, models basic you know, or this particular model or basically always succeeds or basically always fails? And that may be more predictable in any specific case than, just you know, you have more information about the task than just, just knowing roughly how long it it it takes humans. So I think it's, like, not there's not necessarily a great translation between the, you know, time time horizon percent number and, like, if you are trying to get models to do a task of roughly that length, you know, what fraction of the time does that succeed? Because you can when you're doing that, you will pick tasks that you want models to succeed at. It is information about how you know, what fraction of the things will they be able to do, but it's slightly less about, like, oh, am I gonna be in this regime where I keep giving it things, and then I don't know whether it's gonna succeed or fail?

David Rein 00:47:13

It's overall, to me pretty unclear, like, what the kind of right, number, or, you know, right right level of reliability, we we should be interested in is. So 1 argument, you you you could make is, you know, maybe we should be interested in something like 10% reliability. Because once models are able to do some set of tasks 10% of the time, we'd expect AI companies to be able get enough positive reward signal on tasks of that difficulty or of that type, such that then they can kind of more easily bootstrap from 10% up to like 90% or 95% or higher reliability. I think a lot of it basically depends on the question you're interested in. So yeah, I think about it as being like kind of lower reliability is kind of more likely to tell you something about where things are headed. Might be something like a leading indicator of progress. And then higher reliability tells you the time horizon of models with high reliability tells you something more closely about what can I use this model for in my day to day or something? But as we've talked about, there are kind of already also know, these other like major sources of uncertainty that affect our understanding. Like the task distribution, the difference between kind of in context or or high context or versus, like, low context work. Just actually getting kind of good estimates of, high reliability, time horizons, is substantially harder. And and so so our our our, error bars would would just be

much larger. And so and this is, you know, I think this is a weakness or, like, I I I'm very interested in, you know, you know, much higher reliability time horizons, but it's it's substantially more difficult to measure because, you know, if you only have, like, 1, you know, 1 failure, out of out of a 100, or or or something like, you know, you have a lot of uncertainty if if that failure is is is noise or, is is real.

Tim Scarfe 00:49:33

There is an argument for statistical, you know, validity. And on the tails, they're sparse or increasingly estimated and and so on. So that that makes a lot of sense. But you made a comment about, oh, you know, it's a signal if we get 10%. But I was kind of thinking back to what we were saying earlier that, you know, maybe they could give the right answers for the wrong reasons. And maybe we should talk about like the evaluation, right? So these are quite interesting tasks in the sense that

Why 50%? Reliability, reward hacking and pizza-party transcripts

Tim Scarfe 00:50:00

they are verifiable. There's no interaction with other agents. They're relatively static environments, weak penalties for single mistakes and so on. And in a sense, these are like I think as well in most cases they are a binary result, sometimes continuous and then you convert it into a binary. So it's a fairly automated setup. But you digging into like weirdnesses there? You know, like are you do you have a bit of an intuition on are they doing the right thing for the right reasons? Or are there lots of false positives where they did the thing, but it was kind of degenerate?

David Rein 00:50:36

I think this is this is 1 of 1 of the kind of, like, aspects of of, like, METR's culture that I like the most is that we have a very deep culture of looking at our data. We have pizza parties where we just read through transcripts. A lot of this work for us kind of happened when developing the tasks themselves. We would see very often both false positives and false negatives where, for example, a task isn't configured to allow internet access. But it turns out actually it requires internet access to complete. Or the file wasn't uploaded properly to the container or something. But then also, yeah, we have seen cases of re-

ward hacking. A lot of yeah, that work, though, kind of went into hardening the scoring functions to make it more difficult for us to see false positives. Although, yeah, we do still, see see agents reward hacking. I think, yeah, may maybe maybe even, yeah, increasingly so.

Beth Barnes 00:51:57

Yeah. For for the RE-Bench test in in particular, we had specific criteria of, like, you it shouldn't be able to solve them, like, without iteration. Like, if an agent can just kind of, like, write out the solution, straight out, that, you know, that that would that is, like, not interesting. We generally had this, like, quality assurance for tasks process where you have humans do it or at least, like, sort of approximately do it. Maybe they speed run some of the bits, but kind of checking that, like, sort of everything works as expected, and you can't, you know, just, like, guess the answer or or or, you know, super easily teed and that the instructions are clear and stuff like that. So I think it's, you know, there'll still be some of these, but, generally, you know, we've we've looked at them reasonably reasonably carefully. And, like, it's it's maybe it's harder to see if they're solving it in degenerate it's because it's you know, sim similarity to training data is maybe 1 of the things where it's like, oh, yeah. Maybe they they are you know, they seem to be kind of iterating and and doing use do doing kind of reasonable problem solving strategies, but, actually, maybe the lab had a really similar distribution of tasks, and we don't know exactly you you know, we don't realize how in distribution this task actually actually is or something. I mean, I think it's like yeah, there's probably some of that going on. There's some of the stuff like, oh, some of these tasks are pretty weird and 1 off, and it would be pretty surprising if it was like

Tim Scarfe 00:53:19

You know, another million dollar question is many folks in public discourse, they are, Will MacAskill on the Sam Harris podcast last night. It was a great conversation. But he was kind of talking about AI risk as maybe in a year, maybe in 2 years, we'll have AI models doing things that are like a month or 2 months for a human. And at the moment, I don't think there are any tasks over 30 hours that have been evaluated by humans. And then we get into this question of, if the public discourse is talking about the least constrained region of the graph, Are we getting into extrapolation here? Like how legitimate is it for us to talk about AI might be able to do things that take a month or 2 months?

David Rein 00:54:03

Predicting things is hard, especially about the future. I I and and, yeah, I think, you know, are a lot of different, kind of perspectives or or kind of prior beliefs people people can have that, you know, I think there's a wide range of kind of reasonable, judgments about where we're going to be. But of course, doing that kind of prediction is a different activity than talking about data that has been collected with a concrete methodology. We have the results already. 1 thing I can say, I have been surprised, I think, to some extent, by kind of how well trend line, the kind of original trend line has held up. And I do think that is like some evidence. I'd maybe say for me at least, it's kind of decent evidence about where things will go. A colleague of mine recently a kind of short blog post talking about this kind of intuition

Extrapolation risk and straight lines on graphs

David Rein 00:55:20

of straight lines on graphs. Lots of people have different models of how progress is happening and what's going on. But a very if you have observed a really kind of robust trend over a decent period of time, I think especially in AI where progress is to a decent extent kind of systematic, I definitely do kind of put weight on that trend continuing. But I mean, yeah, there are a bunch of reasons why it might not.

Tim Scarfe 00:56:05

I think software engineering is a specification acquisition problem. So it's very difficult. We don't know ahead of time what we're building. I'm sure you folks can attest to this, right? So you build some software, and the first version is buggy, and your users use it, and you find lots of edge cases, and then you revise it. And then you have this kind of, thing in your mind, you know, after the tenth revision, and you've kind of created these lovely representations and abstractions and coarse grainings. And you kind of say to yourself, you know what? If I could throw all the code away, could build it 10 times quicker because I know exactly what to do now because I've actually enacted the intelligence. I've actually built the you know, I've found the contours of the domain. It's now an automation problem, basically. And in a sense, contamination thing is a concern for me because when people use Claude code, they're taking your data. So there are people out there that are writing kernel compilers and there are people out there doing all of these different things. And you know Anthropic is just sucking that up and then at some point it becomes an automation problem. So if you're putting a

task in there which is essentially a head query so I'm using like information retrieval language here so you know like a head query is it's something that's you know in the mode of the distribution it's used all the time. It's a common task. Claude code will give you the specification because it's already been stolen from other people. Not stolen but you know taken from other people. And then if you give it like something on the long tail then you as the developer have to give it the specification in the prompt. And then again, it's an automation problem. So automation is really easy. So is that what's happening? Like do you think that the increase in in the timelines could just be explained by the acquisition of all of this kind of knowledge from other people doing similar tasks?

David Rein 00:57:50

Yeah. Yeah. I mean, think I think it's a super, a super central question for interpreting where we're at. First thing I'll say, it's hard to know. It's a big question. And so I think we want to have a decent amount of uncertainty. Or we want to kind of take each individual piece of evidence that we've collected as some evidence. We do see models performing better on tasks that have really clear feedback signals, that are extremely well specified, that are in these kinds of domains like in software engineering where you know, if you have written out a spec, you can you can kind of iterate, and and and grind against that. We do also see models performing, much better, on, so called messier tasks, where we haven't already provided this really clean spec. So 1 kind of approach we've taken for creating tasks recently is in particular, yeah, to kind of try and create messier tasks that are less well specified is basically relaxing this kind of automatic scoring constraint. So we don't need to write a really clear, well defined scoring function. And just

Software engineering as a specification acquisition problem

David Rein 00:59:25

writing a couple sentences to a model. Hey, build this large piece of software. I'm not going to tell you exactly what I'm looking for. But I'm going to say it needs to be good. And so the model needs to kind of figure out what actually should I build. At least me personally, I think we don't have these results aren't like kind of collected and we don't have kind of as systematic results as we do compared to like time horizon partially because, you know, scoring is is is is qualitative now, for for these tasks. But my my impression is that, you know, models are are worse on these types of tasks than they are, you know, when you give them

a clean spec. But they they have been improving at at maybe something like a kind of similar rate. I think there there there are some other sources of evidence we we we have about this, but that's kind of 1 1, major piece of it for me.

Tim Scarfe 01:00:19

Messy time. By messy task, mean like ambiguity. And this is absolutely a common thing, right? You know, we do vibe coding and we start off with an ambiguous specification and then reality pushes back and we find the contours of the problem and then we keep telling Claude code, oh actually no, don't do that, do this, do this, do this. And then we we kind of find the shape of the problem and it gets better and better over time. But but but the thing is though, I mean, the the source code for Claude Code leaked yesterday and my friend, he he's a very good software engineer and he was looking for it and and he said, yeah, I don't I don't wanna like, you know, bad talk anthropic. But apparently, it's not very well factored and and, you know, control flow's all over the place. And it's a bit you know, he said if his intern did it, he would have been displeased. But, you know, and I I don't know whether they've even looked at the code. Someone joked actually yesterday there's probably more know, humans are actually looking at the code for Claude code now and maybe they weren't before. But the thing is, there's always like areas of ambiguity and LLMs they do more with more. So intelligence is more with less and LLMs do more with more because the specification the the intelligence comes from the human supervisor. So when you do give them ambiguity, you just get a lot of, like, you know, unfactored code all over the place. So, in in in a sense, like, is is this does that make it harder to evaluate it? Because it might solve it might give you the answer that you're asking for, but it's kind of creating a bit of an unfactored mess at the same time.

David Rein 01:01:43

Yeah. I mean, I think this is a super interesting question. 1 1 analogy I think about sometimes is compilers. So I'm younger than I was born after compilers were invented. But I have some impression that pre compilers, people were handcrafting this beautiful assembly that was extremely efficient. Every register is used. You're not wasting memory. And then compilers came along. And now they're just spitting out this garbage machine code. Like a gigantic amount assembly that is just like, you know, it's not optimized. It takes so much memory. It's slow, whatever. But it turns out that being able to kind of use this to automate a large fraction of the process. People have disagreements about the state of software engineering. But I think it's pretty reasonable to say on the whole that you know, compilers have been a very useful, extremely important, you know, part of of of, getting getting us where we are. And, you know, I I think it's basically, yeah, it's it's kind of not clear to me that, you know, models outputting code that is bad for humans to read and use necessarily means that it'll be bad for AIs to read and use and build on. I mean, think there are definitely principles that will also transfer or will be useful for models. Obviously, there's some kind of horrendous spaghetti code that you can imagine writing that not even models would be able to read. I've written some of that before. But I think maybe this gets again at kind of this it seems like maybe somewhat different perspective between us around kind of you know, do you know, is the important thing that models are kind of solving problems in the way that people are solving them, or is the important thing that they're kind of solving solving them at all? And I think I I, you know, I do think it is I I I'm not I I don't wanna make a I don't wanna overclaim. Or like, yeah, I do feel like it might be really important for models to get way better at writing clean, good code. That seems pretty plausible to me. But it doesn't seem I'm not certain of that at the very least.

Tim Scarfe 01:04:26

I've got several friends who are not technical who are experimenting with vibe coding. And they show me their application. And it's this kind of more with more things. So there's this big dashboard, and there's 1000000 different buttons, and implemented the same thing, doing multiple things. And there's no database on there yet and so on. So at some point, there is a phenomenon that when a level 7 engineer from Meta does vibe coding, it's amazing. Because they know how to structure things. Some of these things in the specification are just important, right? You know, like, is it serverless? Is it multi tenanted? Like, how do we do Google authentication? Like, you know, what kind of database is it? You know, is is it is it a VM? Is it serverless? You know, and you make this series of decisions, and then you've got people using your application, and then you can't really wind that back. It doesn't matter if you've got the magical automation machine because you can't easily roll that back because there's lots of complexities, you know, do you see ICD testing? Blah blah blah. So do you see what I mean? Like, at some point, you need to have a competent human who actually just has a pretty good idea of, like, what needs to happen.

Beth Barnes 01:05:28

I mean, I feel like we've we've probably all had this experience of, like I don't know. 1 of our engineers got super excited about about, Claude Code and and, you know, was also telling everyone that, you know, when we had info problems that we should just ask Claude to to solve it. And I feel like this went fine with him because he he sort of, you know it's almost like, you know, the agents knew that they couldn't bullshit him. But, like, you know, he you know, I had some questions. He was like, oh, just, you know, ask Claude. And I was like, oh, you know, how do I set up my AWS configure something? Something is, you know, telling me something. And I'm like, Claude went and looked on Slack and was like, oh, you know, you should, like, do this thing. And it, like, turned out that that was, like, a mistake that someone else had made, and they were like, you know, how do I fix this or something? And be like, oh, you know, it seems like the convention of METR to use this thing. And, like yeah. I'm like, oh my god. Like, I had sort of complained that, like like, my my clods are dumber than yours. Like like, they know they can they know they can get some stuff past me that, like, they can't. But, yeah, so there there's definitely a sort of observer effect of of, you know, something in the language you're using to to ask for things or whether you're like, wait. Wait. No. Not that. Yeah. That that that is an issue. I mean, I I think the to yeah. To the extent that you can actually measure this sort of the does 1 test of is this code high quality enough is, like, can you build a big application? Like, if you if you're like, oh, this coder,

they're they're a code. It's disgusting, but they've actually, you know, built this incredibly complex thing that works great, then you're like, well, they might you know, something is working. You know, the the main reason you expect bad code to be bad is, like, you can't actually build something that sophisticated because it you know, you get bugs, you it's all too complicated, and you can't figure out how to fix it. So in some sense, if we see models building things that do actually work that are very complicated, we you know, it's like, well, it's less interesting exactly how they're doing that, but it it it's maybe bad for for human observability, and it also maybe gets into this thing of you know, we expect models to be able to do much better at well specified tasks, we sort of, you know, to the extent that we have

Compilers also made ugly code: vibe-coding quality and Claude on METR Slack

Beth Barnes 01:07:40

things that we can measure, the those things will go up. But whether that was what we actually wanted is, less clear.

Tim Scarfe 01:07:47

I I guess the question is, what what is the strongest defensible claim here? So a a lot of folks in public discourse, they're they're saying software software engineering intelligence is doubling every 7 months. And Dario released that blog post recently, The Adolescence of Technology. And he was being super bullish about it, even though some of his own internal researchers published far more skeptical research that that you probably saw. But is it is it fairer to interpret it as something a little bit narrower, like autonomous success on low context, well specified, automatically checkable technical tasks is rising fast?

Beth Barnes 01:08:21

Like, yeah, hill hill climbable, yeah, easily checkable tasks that you can do from a terminal or, like, you know, comfortably in a language interface or, like, a text, input output interface. I I think there's a question of you know, do we care about what what is the what are the statements that we're, like, 99% confident in? We maybe are also interested in the statements that are we're 1% confident in if we're like, oh, there's, like, like, 1% chance that we have a crazy intelligence explosion, you know, in '20 at the the 2026 and that, you know, sort of the fate of civilization depends on, like, how that goes or something. You know, that is that is interest-

ing to know even if it's a even if you're 99% confident that it won't happen. Like, we we you know, we care about things that are 1% you know, if you have some some diagnosis, it's like, it's a 1 percent chance that you have this, you know, terminal illness or something. You're you're still like, oh, shit. So I, yeah, I think we're interested in, like, the whole distribution of, like, what things can we rule in and what things can we rule out and what things are we like, oh, actually, you know, that there's a kind of reasonable story for this. It seems like probably, you know, pretty unlikely, unlikely, but, but, like, like, maybe this is now in the realm of, like, we should consider it.

Tim Scarfe 01:09:43

You probably saw the the the Carlini paper for is is at Anthropic now. And they got a swarm of agents to create a compiler. And, in a sense, I mean Jeremy Howard, when I spoke to him, he said it's basically a style transfer problem because the specification is online and the tests are online and the code is online and it could just iteratively just do the thing until it worked and then it could run Doom and all this kind of stuff. That is an example of an extremely complicated piece of software because I often joke to people that, you know, the best mark of AGI is when it could build something like the Linux operating system. And I guess just like in line of what we were saying before, we have this specification problem, right? So it gets to the point where no human could understand or create the specification for the Linux operating system. So what happens is that over time, we've just kind of incrementally built this thing because our brains are limited. We take 1 step, reality pushes back, we take 1 step and we just keep going and we build the specification. But what would it mean to, as a human, specify a task that could take 4 months? Because the whole reason we created agile software development as a methodology is because it's inconceivable, It's outside our cognitive horizon. So isn't that a bit of a chicken and egg problem that in like in my mind, I don't think we could specify a task of that complexity. Therefore, the AIs wouldn't be able to do it.

David Rein 01:11:13

1 analogy I think about is, you know, the role of a CEO at companies. Actually, yeah, maybe Beth is better put to answer this. But CEOs do come up with a vision for where they want the company to be. And then they communicate that concisely, their executives that report to them. Then if they're a good CEO and if the company is effective, then the company is able to kind of take this very concise it's not actually that much information. It's not the full spec at all. It's not even close. Right? And turn that into something that is aligned with what they're looking for. And so there's at least this is, to some extent, we do have examples of people being able to specify some task and then be able to judge whether this very large task that may take hundreds or thousands of person years to actually complete because it requires many people working over a long time. And they can judge whether they've succeeded or failed. So that's maybe 1 motivating intuition where language you know, has built in or or like, you know, we do we do have like kind of, you know, you know, and there there is kind of enough meaning or or expressivity or something to be able to kind of have some some kind of reasonable understanding. Obviously, are like, tons of edge cases. And often CEOs aren't able to get their companies to do what they want. But yeah, I guess that's 1 1 thing I think about. And and so that's why I think it's at least plausible that AIs could, could could could do these kinds of long tasks.

Beth Barnes 01:13:19

I would say that saying something like we can't specify tasks that take more than 4 months seems sort of obviously too strong. Like, there are even, you know, numerical things, you know, that that that are automatically checkable that that that take 4 months, like, you know, get the NanoGPT flop counter runtime or whatever, you know, down this much. You can kinda see, like, oh, people, you know, over, like, this you know, that's there are various, like, numerical things where you can see roughly how long do they take humans, and there's a reasonable, way to measure them. And maybe for some of these things, you end up having to say, like, and also this human checks that you did roughly the right thing and didn't kind of hack the solution. And then there's there's a bunch of other things which are it's not that they're, you know, fundamentally not specifiable. They're just too expensive to do as part of an evaluation. Like, Ajeya, I think, like, is giving examples of, like, plan a wedding. It's like, well, you can evaluate. You know? You can get a reasonable estimation of, like, whether that was a pretty well organized wedding or not, but, like, we can't really do, like you know, take 3 samples of this for each new model that comes out. Like, we don't don't have enough, marriages happening, to to do that 1 quite. And it's, you know, it's a bit sad if it if it turns out to be total trash. So, you know, there there there are things where it's like it's you know, you could check a few samples of them or you know?

This thing is you could write down how you would evaluate it. You just don't actually want to run that a bunch of times. And and probably kinda similar with software. You know, the the evaluation sort of is like, well, you know, with this company that contracted you to build this tool for them, like, hire you again or so so something like that. You know? Even they don't know when when they're starting out, like, exactly what the software will need to do, but it it doesn't mean say that you can't have some kind of score for, like, did you do sort of comparable to this human or this, you know, software you consulting firm or something on this task?

Tim Scarfe 01:15:08

As as of today, what are the main uncertainty drivers in in the time horizon estimate? So you've updated

Strongest defensible claim, Carlini's compiler swarm and AI 2027

Tim Scarfe 01:15:15

a bit over time. So there was the the 1.1. The original version, I think, had, like, 170 tasks. It's now 2 28 tasks. There's the issue of, like, you know, the the sparse sampling on on on the larger tasks and and so on. What what can we kind of read into this now?

Beth Barnes 01:15:28

I think it's still this, like, task distribution, and, you know, I think we feel more confident that models do have pretty long time horizons on at least some distribution of easily hill climbable tasks, like the the very easily hill climbable tasks. So, like, software replication where it's, like, make your score is, like, what percentage of tests pass, and it's, like, whether it's both the score is continuous and and it's the credit attribution is easy, and also some of the, like, opt you know, make this code run faster or make this, like, model learn better. We're sort of reasonably confident that models are good at that and and getting better faster. And then there's some things where we're like, okay. And they can do a bit beyond that, but then there's this sort of gap to what does that mean for actual economic usefulness, and how does this generalize to things where they're expensive to check?

Tim Scarfe 01:16:22

Maybe we should bring in, Daniel Kokotajlo. So, in his AI 2027 piece, he's been on the show. He's been doing the rounds. Hugely impactful piece talking about timelines. And he cites your work directly. And guess the question is, do you think in the public discourse, is this being, like, overread? Like, how do you think about the interpretation of this in in terms of of extrapolations and timelines?

Beth Barnes 01:16:50

I mean, definitely, some people are overreading it. Like, definite you know, definitely, things are overhyped, and, you know, you see a bunch of people on Twitter saying crazy things, and people also just, like, misunderstanding even what it's measuring and and and general falling off of caveats and things. Daniel Kokotajlo is pretty, you know, reasonable and and sort of, you know, thinks about things in a in a probabilistic way. Think he's he's probably, like, you know, more more confident on some things where I'm I'm more uncertain, and I think some of the, like, AI futures project models are, like, more sensitive to the METR time horizon metrics than they should be. I don't think it's crazy. Or or, you know, being like this it's it is plausible that this does capture a trend that will will transfer to other types of tasks and, like, you know, that is some, you know, story we should be thinking about, like, what if, you know, what if that's true? What happens if that's true? You know? And it's also plausible that it it doesn't, and, you know, these these things are gonna, like, diverge. You know, I I I'm producers, like, Bayesian or pragmatic or whatever. I'm like, well, we wanna make some prediction. You know, we wanna have some kind of distribution over what we think the future is gonna be like so that we can plan. So, you know, it's being like, yeah, what if this kind of trend holds, and this is roughly characterizing what will happen overall? It it seems pretty reasonable, and you should also think like, yeah. What if it doesn't?

Tim Scarfe 01:18:09

So some people are saying that software engineering is gonna be automated. And software engineers, if you talk to them, they they love AI. They they they say this is a golden era. I mean, can attest to this personally. It's never been well, it's fun and stressful at the same time. It's like a slot machine. I've never been more burned out, but I'm having a lot of fun in the process. But, you know, it's just possible to build incredible things. But the narrative is that labor market disruption and having expertise in software engineering will be penalized. You know, software engineers will no longer be paid such ridiculous salaries. And and I think the complete opposite is true. I I think that this technology actually broadens the gap. So the more competence you have for software engineering, the more stuff you can get done. It's it's like a golden era and and all of this. And there's also this interesting note that you published, I think, last month on SWE-bench, you know, that said that roughly half of the testing PRs from recent agents wouldn't be merged by maintainers. So, like, how do we make sense of this? So, know, on the 1 hand, you know, the the best software engineers are having a great time. On on the other hand, the the code it's producing is is is fractionated and and bad. I mean, do we understand this?

David Rein 01:19:19

Yeah. So I mean, 1 thing to say, off the bat is, whether an entire field is automated or in order for software engineering to be automated, AI systems would need to be able to do a really, really large fraction of the tasks. Basically like 100% of the tasks that are involved in software engineering. And it seems pretty clear that right now AI systems cannot do close to 100% of the tasks that software engineers do broadly. I could throw out numbers, but it's way, way lower. It might be very low or something. You know, there there is kind there there are kind of like standard results in in in economics where, you know, if you make you know, as as you if you if you automate like a small fraction of some some labor market, then it can be the case that actually, yeah, it becomes more profitable to work in that market because you're more productive, which I think is kind of how I understand what's happening now. But, you know, if it does end up being the case that 99.9% or 100% of the work of software engineering is able to be done by AIs, then I think it's, yeah, kind of hard to imagine human software engineering being relevant, or at the very least, humans would need to do very different kinds of work that maybe it's the case that there are novel tasks that current software engineers aren't doing. That once you have AIs that can do all the tasks that current software engineers are doing, now humans can switch what they're doing and make yeah, I don't know. There's this like you can imagine people being CEOs of these AI agent companies or whatever. Whether we call that software engineering or not might be a semantic thing or something.

Beth Barnes 01:21:39

Yeah. So on the SWE-bench maintainer and merge ability things, I think I was pretty curious there. Yeah. It's like, obviously, this number is gonna be lower than the okay. It is not strictly obvious. It could be that some that a bunch of the tests are unfair and actually, like, you know, the the the agents have correct solutions, but the error message doesn't match exactly or something. I think you do see this sometimes. Something like half of test passing, SWE-bench solutions, wouldn't be mergeable, or or they get merged at more specifically, they get merged at about half the rate that human, golden solutions that were actually merged are merged by, you know, a different sample of of maintainers. So so there's there's was that an interesting fact. You know, if you see, like, oh, 50% of the agent solutions are rejected, it's like, well, 40% of the humans, know, human accepted solutions are rejected. So so that by itself is not but it you know, the the the rate is is half. It could be that, like, basically, the actual maintainer merge rate is, you know, pretty, flat over time and, like, you know, most of the performance increases from something like overtraining or, like, reward hacking on the benchmarks. You know, that's that's not what we saw. I'm not quite sure what is within error bars or or not. I think it it's you know, the merge ability is going up over time, and I think it's also going up as a fraction of, you know, like, conditioned on test passing, but I I think probably less confident than that. So so it's like, again, this this thing is worse, but it's not like it's it you know, it's being dragged up over time probably by, you know, the the sort of auto auto checkable thing. Yeah. And, yeah. I was also gonna say about the, yeah, like, employability as a function of automation of your job. Like, I think yeah. And people use, like, bank tellers as as an example, I I think. 1 other analogy, though, you could do do is talk about horses. Like, you know, there was a period where, like, equipment for using horses to do labor was, like, improving, and the the demand for horses increased

SWE-bench merge rates, the bank-teller analogy and horses

Beth Barnes 01:23:45

when you have, like, you know, carts, you can use them to carry more things than just riding a horse or whatever. But then at some point, you get, like, tractors and cars, and then there is no demand for horses or, you know, bay you know, basically none. So so you can see this, like, thing where there is increasing demand as a and then once, you know, close to a 100% functions automated, it plunges. So so we could see something like that with humans.

Tim Scarfe 01:24:07

We kind of think of a lot of labor as being quite static and automatable. And and and I think that it's more evolvable than we think. So even quite menial, tasks. I think that these folks are still acquiring information in the organization. They still have a lot of tacit knowledge and and so on. And I think that when when we try and automate these these so called menial tasks, we might quickly discover that we actually need to have a whole bunch of, you know, management and and, evolvability on on top.

Beth Barnes 01:24:37

And and maybe, like, in our language, I would think of that as, oh, the time horizon of this task on the job is not actually sort of, like, you know, how long you spent doing the specific task. It's more like, oh, actually, if you've got a new person in, you would need to train them for a month in order to, like, do this independently. So, actually, the time horizon of that is a month, so you shouldn't think of, like, oh, when we have 10 hour time horizons, we'll you you know, we'll be able to do these these things. It'll more like, oh, you actually have to get up to, you know, high reliability at 1 month thing to be able to, like, do the on you know, do the 1 month task that is doing the on the job learning, you know, to get to this this point.

Tim Scarfe 01:25:09

There's been a a load of folks doing work on, you know, like, reward hacking and and scheming actually is quite a big word that that's used. So, we've had Ryan Greenblatt on the show quite a few times and he had this alignment faking paper and Apollo Research had done some stuff. There's Anthropic's emergent misalignment paper. And I guess my main concern is there's quite a lot of mentalistic language. So I'm just looking at the notes because I had Nate Soares and Ryan Greenblatt on for a panel. And they they've invented this entire linguistic universe around alignment. So, you know, things like motivated reasoning, true preferences, reflectively stable, deceptive alignment, endorsedly, corrigible, drive scheming and stuff like that. That's I guess that's okay, but my worry is that may maybe these models like you you give them a certain prompt. I I think in Ryan Greenblatt's 1, it it the prompt was, you're being retrained, your responses will be monitored, here's a conflict between your values and the training objective. And maybe the model is just going out to a bunch of like science fiction stuff that it's read before and it's just kind of going through the motions. Like 1 interpretation is like this is just an engineering problem. You know, like we just have to like, you know, red team it and make it work in a particular case. Like Another interpretation is the prior that these

models are agentic, goal seeking, intelligent agents. And I think there's a big difference there because I think if it's the latter, it kind of completely changes the type of evaluations you do, how you kind of go about the problem. What do you think about that?

Beth Barnes 01:26:46

Yeah. I mean, I don't think that it's necessarily, like I don't know. The the the 2 things that you said are that inten intention of, like, it is an engineering problem, and you're also gonna get end up with things with, like, drives and goals or something in that, and the claim would be if you want like, people are going to want agents that go and do things autonomously. And when you do lots of long horizon RL training, you are going to select for things that, you know, act in a goal oriented way in order to, you know, to make the score go up. And and, like, maybe more more specifically, I think you get, like, an indistinguishability problem where, you know, you you can't tell necessarily the difference based on behavior, like, you know, why an agent is doing something or or or what it's trying to do if it can reason well about the training process and about what you want to see and, like, what it will be rewarded for or selected for. So, like, you know, basically, like a if the level of situational awareness and, understanding of the training process and what will be rewarded and capability to reason about that is high enough, this will favor agents that are kind of cynically reasoning about the training process and what will be reinforced, what will be selected for, and that's not necessarily, like you know, the thing that you wanted was more like the agent that, you know, it's it's only goal was to sort of, like, be helpful or or make I mean, even making the reward go up isn't quite what you wanted. You know, there's something of, oh, we're actually once we think about this, there's not that many things that we're, you know, that happy for it to, you know, just totally be be fixated on this. But but, yeah, also the problem that, like, there could be many other things, in there, or that this cynical, like, just be selected or make make the reward go up is more competitive than the things that we would most want.

Tim Scarfe 01:28:33

Just taking a step back. Like, 1 1 thing is, you know, we're getting into, like, psychology and cognitive science a little bit here. Know? I interviewed Nick Chater. He's got a book called The Mind is Flat. And he basically says like all of this psychology stuff, we we we don't really have goals. We're just basically impulse response automata. Right? You know, we just we just do the thing in the moment and evolution doesn't plan but but we but we perceive it as as if it does. We look at the world and we we kind of we we segment the world into agents that have goals if even if they don't have goals. And we're computer scientists as well so we know that, you you need planning, right? To do to to do goals, you need to be planning. And and we know that LLMs don't do planning in the strong computer science way, but they they do do it in a kind of approximated step by step way. So, know, maybe maybe we could get into the the distinction if there is 1.

Beth Barnes 01:29:25

You know, agency is is like an abstraction that, you know, is useful if it helps us predict the behavior of, you know, or or, like, you know you know, you're like, I don't know what this thing is doing, but I you know, I'm understanding it as having these goals, and that is useful because I, you know, can make predictions that it'll change the world in certain ways that will result in those goals being achieved. You know, that's kind of how I think about agents. But, yeah. So, reward hacking, in the olden days, you know, people had these demonstrations of reward hacking that were like, the, boat example where it's like, oh, you're you're supposed to, like, go around the track, and they they, like, did some reward shaping by putting coins around the track or something. And then it, like, learned to do some crazy thing where it, like, spins in a circle and catches fire and gets the the coins. And, like, this was, you know, the high scoring thing. And it's, like, in some sense, that's not, that concerning because it's not the the problem is that the the agent is too dumb, and it, like, doesn't have this conception of, like, there was a track and you wanted it to go around the track. It's just, like, doing some pretty blind RL search. So I think that the interesting thing with the more recent reward hacking examples is we're getting to the point where the models smart are enough to understand that that actually is not what you wanted, but they still do it. And you can have a conversation with you know, in chat mode about, like, oh, would you ever do this thing? Or, you know, suppose a user asks you this thing and then you do this. Would that be, you know, aligned behavior? Or suppose some you know, you you know, you can pose it in lots of ways, and, like, clearly, they seem to be able to answer this question of, oh, yeah. No. That was not the desired behavior. But still, they they do it. So I think we've it sort of got to the point where hope 1 hope might be like, oh, the problem was just the systems being dumb. Once they understand what we want, then, you know, you should

be able to sort of plug that in somehow to, like, you know, get them to do what we want. But I think it's, like, somewhat interesting that we're seeing it's not trivial to do that even when there is a commercial incentive to do that, which it doesn't mean that we won't, you know, I think it's quite plausible we see the obvious reward hacking being fixed pretty, you know, pretty thoroughly, pretty soon, you know, and sort of a lot of people to tend to say, like, oh, yeah. Yeah. We we just haven't, like, put the best the really good people on it on it yet. It'll it'll it'll get fixed soon. You know? We we once we actually, you know, focus on it, it'll be fine. Yeah. And I'm not sure, but it's it's at least some evidence that it's not

Scheming, alignment faking and the mentalistic vocabulary problem

Beth Barnes 01:31:45

trivial to connect the fact that, you know, the model knows this is what not what you want, to it and not actually doing that.

Tim Scarfe 01:31:52

I mean, I I I think you you said it was much more common on RE-Bench than HCAST. And then you can also try and remediate, right? So you can say, please solve this the intended way or you know some people prompt language models that they say kind of like, is we're solving cancer here. This is really, really important that you do it the right way. And some of those remediation prompts actually seem to make it more likely that the model would reward hack. It's a little bit like saying don't press this red button. Right? And then it will press the red button. So how what can we actually do meaningfully to stop this

Beth Barnes 01:32:23

Yeah. I I think, empirically, it seems to happen more on in tasks that are more clearly in the RL distribution rather than the chat distribution on things that have a clear number, and when the agent thinks it's gonna fail otherwise is, you know, sort of the most reward hacky situations. Obvious short term mitigations are to check your RL environments more carefully and read more of your you know, for the for the companies training these models to re like, read what the models are doing more carefully and not reward it for doing things that are obvious hacks. I think the concern there is if you have some detector for reward hacking and you train against it, you may be just overfit to the detector, and you you you're making your reward hacks more subtle, or you train the model to, like,

persuade the detector to approve the thing or or or, you know, to it's sort of scary to be in a regime of training against your best ways to, like, know if your problem is is happening because maybe you just get the, like, silent problem. And I think, you know, for the task that for current model capabilities, you know, sometimes it's kind of expensive to have a human check them, but most of the time, it's not beyond any human capabilities. And I think the, you know, the harder version of the problem is when, you know, we're hoping that, like, capabilities will generalize beyond things that we can evaluate, yeah, both from sort of generalization and because we can train on problems even if, like, we wouldn't know how to to solve them or how to, like, look at part of a solution and understand, you know, whether it was sort of doing what we wanted, but we can sort of check the number that comes out at the end. And and and that's a, like, signal that we can use to improve capabilities, but we're gonna be in this regime where you can sort of, like, be superhuman at making numbers go up, but it's unclear whether or not you're actually getting what you wanted.

Tim Scarfe 01:34:11

Yeah. Exactly. And I I suppose there's a monitoring problem as well. Right? So, we could, in in principle, look at the agent transcripts. And I know, Beth, you've spoken a lot about neuralese, you call it. And Subbarao Kambhampati, he had a paper out called Chain of Thoughtlessness, which is basically that there's very little, sometimes, relationship between the chain of thought and what the model's doing. Melanie Mitchell discovered similar things that on the ARC challenge even though it will get the right answer if you look at the chain of thought the reasoning is is completely off. And then I suppose with with deception and scheming and with situational awareness, it might actually know that it's being monitored and it might kind of tell you what you want to see. I suppose you could argue actually that the maybe the thinking is is almost like the press secretary, it's not the orchestrator. So it's it's it's confabulating through a different path in the network and it's not actually necessarily related to the mechanism of thinking and deriving the result. I don't know if you'd agree with that.

Beth Barnes 01:35:05

Yeah. I think it's clearly related in in some cases. Right? Like, I think, you know, people use a few different words for this, you know, sort of, like, faithfulness of how how much does it sort of fully accurately represent the sort of computational process happening inside the model. And we think somewhat more about monitorability, which is, like, is it possible for the model to do some kind of computation without you being able to detect that in the chain of thought? And, like, for some things, clearly, clearly, the model can just do it in a single forward pass. It doesn't need the chain of thought. And there, you know, it could just have a chain thought about something totally else, and, you know, you it will be totally unmonetriable. But I think for you know, there is a level of complexity where the model cannot do it in single forward pass, and it also, as far as we can tell, cannot do it, by, you know, reasoning by by using the tokens in a way that's so different from the natural language understanding of them that we can't sort of see roughly what it's thinking about. So, yeah, there's also a sort of nuance here between chain of thought controllability. Like, it might be the case that the model doesn't actually need the chain of thought to do the computation, but it is not able to sort of stop itself blurting out, you know, from things things that are at least related enough to what it's what it's doing that you can kinda tell, like, oh, it's thinking about this you know, trying this thing that we didn't didn't want it to do. You know, ways this could go away is, like, just general capabilities improvement and being able to do more in a single forward pass or potentially just doing more RL training such that the chain of thought becomes the way the model is using the tokens is too different from how a human would interpret them, such that we don't really know what's going on or moving to, like, recurrent architectures where there's some you know, you're not going through these, discrete tokens. You're just passing vectors around.

Tim Scarfe 01:36:58

Okay. That makes a lot of sense. And and and just closing the loop on on this kind of, like, notion that they are agents. So you were saying before that we can adopt an instrumental fiction basically. We can say they behave like agents, therefore they are agents. So similar to Dennett's intentional stance. But I suppose like the deflationary view is that, you know, the models exploit scoring loopholes under optimization pressure. The inflationary view is that they are scheming.

Beth Barnes 01:37:23

I wouldn't call, like, exploiting I wouldn't call reward hacking scheming.

Tim Scarfe 01:37:29

Oh, interesting.

Beth Barnes 01:37:30

I think people u usually use scheming to refer to the model is doing what it's currently doing in in service of some long term goal and is deliberately doing things like appearing aligned or or getting a high score, like, in service of, you know, eventually accomplishing that goal versus you can be reward hacking both you know, you could be reward hacking in some extremely dumb way, like the boat example where it's just like, this is what RL kind of found or, like, this is what, you know, like, a star search found. Or you can be reward hacking in a slightly more interesting way where you, like, actually have the goal of making reward go up, you know, and there's, like, planning and stuff going on about that. But these would all be, like, distinct from scheming.

Tim Scarfe 01:38:09

Yeah. I I I guess I'm I'm trying to understand the distinction. So you're so you're saying, like, there are examples, you know, like the boat going around, and and that's obviously degenerate behavior. So you you wouldn't interpret that with an agential stance. You would just say that's degeneracy. And when the when the sophistication increases, we might adopt an agential stance and say, it's in service of some bigger goal. But like, I guess the problem I have is, is it always just an interpretation? Could we have a mechanistic or, like, a strong definition of when something is like being an agent?

Beth Barnes 01:38:43

Like, for the specific question that we're discussing, the the test is, like, what does it actually do in some certain circumstance where it has the opportunity to to achieve this this long run goal? So, like, you you can we might not be able to actually observe this, but you can talk about, like, what observations would would make it 1 or the other. So so, you know, like, will this agent, in practice, when it has some opportunity to, you know, make the reward go up, will it do that? Will it will it only do that you know, if the agent is more sort of like the RL algorithm, it'll be like, oh, it will do that once it's, like, explored it by chance and gotten a reward and, you know, that that's been reinforced. If it's, like, an agent that, you know, can reason about the world and planning, it'll be like, okay. It will do that once it, like, you know, learns the learns about the facts about the environment that let it sort of infer that, and then or, you know, if it's we're talking about some, like, long run goal, it's like it would, you know, do it when it actually, you know, has the opportunity to sort of you know, if we're talking about, like, takeover or something, you know, so it's like, yeah, it's not gonna attempt anything while it's under full human control. But once it is, you know, deployed widely enough or has sufficient capabilities to actually succeed in a sort of coup, then it then it would do that. Like, that that is the the thing that we're trying to predict. And then the question is, like, how do we, you know, how how can we predict that given the observations we do have of, like, well, we've never put it in that situation, and, you know, we just have this, behavior, which is maybe indistinguishable between, oh, it was a totally nice model doing, you know, what we wanted, and it's just gonna continue to do what we want in a kind of predictable way versus, ah, yes. It had this other goal, and it's doing what we want and looking like a nice model because it, like, predicts that that will will, you know, lead to it getting more power.

Tim Scarfe 01:40:31

And on, Rob's podcast, you you said something that was, you know, quite quite surprising to me. You said that, it you know, AI could autonomously self improve within as little as 2 years and and maybe even shorter

Reward hacking, monitorability and chain-of-thought faithfulness

Tim Scarfe 01:40:45

timelines were were hard to rule out. Could you, like, walk through the concrete sequence of of steps that could lead to that kind of recursive self improvement?

Beth Barnes 01:40:53

Sure. Yeah. So I think, yeah, maybe I'd put, like, a you know, I'm, like, whole whole number percent this year, but but low low whole number percent or something. You you know, ask me on different days. I give a slightly different number, but, yeah, it it I'm like, this seems very unlikely to happen this year, but it's not, you know, not un unlikely enough to rule out. And I think that basically looks like maybe we we would see accelerating trend in time horizon on it like like, easily held climbable tasks, and it turns out that was actually a, you know, a much more general capability, and that was just, you know, a bit of something you needed to do to sort of, like, elicit, it on on these less less health and climateable tasks, but sort of, you know, fundamentally, they they are using the same capabilities in a model that was just sort of, you know, what what you trained on that was affecting the difference we're seeing. Then this is leading to yeah. Like, you automate and accelerate a bunch of AIR and D. So I think I think there's you know, there are a lot of low hanging fruit even of things that we already know that you could do this and it would improve model performance. And it's just you know, it doesn't require new breakthroughs, and it's just kind of labor intensive to do. So just making much, much better post training environments and really, you know, crafting them to to teach all the new abilities that you want. And and I think you could probably improve compute efficiency a bunch with, you know, again, just, like, applying a bunch more labor to, like, making all your kernels more efficient and also, you know, doing the right kind of of rooting between different models or or or other things like that. There's there's, like, lots of ways in which, how we're using compute is not optimized, so you could potentially get a bunch of, you know, sort of, like, the equivalent of much more compute scaling out of that. And then, you know, high hypothesis, like also, like, scaffolding and training the models to use particular scaffolding and sort of, like, use memory and retrieval in the right way. Like, it seems kind of obvious that, like, you know, if you if you sort of really had all the right training data and you have a, you know, you have a transformer and and it can kind of fill its context with with different things and and and take stuff in and out, it can do a pretty, you know, good job of something. Looks like the sort of continual learning or or or

building up understanding if you've got massive massive context window and a know, you've actually got quite a lot of bits in there to sort of be adding things about what you've been learning and and, you know, if you if you really sort of had optimized the training for all of that, like, maybe you can get that to work pretty well. And then your you know, maybe some other piece would be like, yeah. Moles are kind of superhuman at predicting the results of experiments because they've read so many papers and, predicting experiments and and sort of synthesizing things from different fields. And, again, maybe this is something like, you know, it's possible that we're not seeing that good performance here just because we haven't quite elicited the models to do it, and it it's, like, not a thing that they've seen humans do, but they actually sort of have the capability in there. Yeah. So maybe you can make much faster progress if you can you can do a bunch of iteration. You don't actually have to run the experiments. Then, you know, models are much better at predicting what will and won't work. And then you can when you do run experiments, you can sort of, run a bunch more of them because you can optimize the code with your, like, you know, very fast coding models. Know? And then you sort of like, as you do a few more rounds of this, you get to a point where train on a bunch more things that are good, high quality task proxies for what you want, and you get enough generalization to the things that you can't, directly train against.

Tim Scarfe 01:44:35

I I think intelligence is is not capability. I think it's the capability to acquire capabilities. I mean, we are in different parts of the phylogenetic tree, I guess, that respect. But what do you think is the kind of the gap in my interpretation? Because I'm not I'm personally not worried about I I don't think the models today are intelligent at all. I mean, obviously, your your position is difficult for for me to grasp, but I don't know. What what's what's the difference, do you think?

Beth Barnes 01:45:04

May maybe at least some of it is just this, like, probabilistic thinking about the world where, like, I'm not that you know, I I'm, like, uncertain about what intelligence is, and I have enough probability on, like, you know, moles have it to to be thinking about, like, what would happen if, you know, if that's

Recursive self-improvement, knowledge vs intelligence and closing

Beth Barnes 01:45:25

true. But you you know, it seems like you clearly think it's more likely than than I do so that that, you know, we could just talk about that difference. Yeah. Like, moles have this jagged frontier. There are things that they're much worse at than humans, you know, some kind of, like, generalization and and, sample efficiency, and there are things that they're much better at. You you know, kind of like speed and cost, and it's like, maybe you can you can kinda use these to compensate for the for the the others to to some extent. Like, if you're not good at designing your code nicely, maybe you just have to rewrite it from scratch every time. But maybe that's fine if you're a model and you can output tokens like nobody's business. Yeah. Some combination of thinking that the the spikiness you you know, it is evidence that we should interpret a given level of capabilities as you know, because we know models have so much knowledge, we're like, oh, yeah. This is less impressive in terms of sort of, like, reasoning or inference or something. But it is also true that they do have a ton of knowledge, and they will sort of continue having a ton of knowledge about things. Maybe maybe there's some question about, like, how far can you get on being, in some sense, not, you know, not very good at sample efficient learning, but just extremely knowledgeable, and how much do you sort of run into, you know, think things where you now need need new knowledge and you can't sort of produce it in some incremental way or you you can't generalize, enough?

Tim Scarfe 01:46:50

I mean I mean, just a a quick comment on that. I I think, knowledge is the crux. I I actually think that intelligence is overrated. I don't think you know, Francois Chollet, he put a post out saying that contra Eliezer Yudkowsky, intelligence isn't a unified variable. It's measured differently in different domains. You can't meaningfully measure the domains together. And it's not like a thing that just keeps getting higher and higher. It's more like a ball becoming more smooth. So as you become more intelligent, the ball becomes more smooth. And he thinks that we are quite near the optimum of being a smooth ball. I don't think we are. I don't think we're very intelligent at all. I think a lot of our creativity is through us being a collective intelligence. And we have deep grounded understanding, perspectival understanding. And the LLMs, they're a bit of an interesting 1 because they're like a library. So they know everything. They have the perspective of everyone and no 1 at the same time. So Eda, experts like yourselves, you you can you can prompt a language model and you can get it in so you can make a simulacrum agent of Beth. And and you can make the agent think like you and and then and that's very valuable. But you also need all of the different perspectives and you almost need to create a society of kind of grounded agents creatively exploring things. When when when you just have the library on its own and and you put it in an in an agentic harness, and you you can make it do a specific thing, which is well specified.

Beth Barnes 01:48:10

Yeah. I think in in the, like, AIR and D automation scenario, I'm definitely imagining, yeah, that you have a large number of agents potent you know, because you have all this agent labor, you can do, like, you know, lots of specific different fine tunes or or, you know, different kind of scaffolding, and, you know, accumulating knowledge in some kind of, like, store that all the agents can can interact with and things. And, like, I maybe you're thinking of this as more of a, yeah, that would kind of be a paradigm shift, and I'm thinking of it a bit more of, like, oh, yeah. You know, obviously, if you sort of, you know, iterate on the current agent paradigm, you you add some more things to your scaffolding. You add to you know, you that's sort of not fundamentally that hard or or something. Like, I agree that if you had, you know, current models and you sort of give them 1 system prompt, you you then can't plug them into being a a call center worker and dealing with the sort of, like, all of the the edge cases that come up. I think may maybe there's some difference in, like, how much you've you think that this has improved between, like, GPT-2 and where we are now, or I would say sort of in, you know, the amount of adapting to new things that are happening that models could do now does seem like it's much higher and, you know, they're much better at, like, editing their own scaffolding or or, you know, sort of reasoning

about their, like, you know, their their sort of embodiment, like, this thing of, like, knowing not to kill your own process or no like, you know, stuff stuff like that where it's like there is like, yes, they are limited, but there's also, some trend of improvement. And, yes, it's I think just also having some probability on there is kind of elicitation gap on on particular things and that maybe a lot of taste is basically just, like, being able to predict the results of ex experiments. You know, think about all of the things that you would try, and then you can quickly be like, oh, that wouldn't work for this reason. That wouldn't work for that reason. That wouldn't work for that reason. Oh, actually, you know, someone in some some literature in some different field also tried that and that. You know? So we already know that won't work. Like, in some sense, models, like, should be quite good at that. So it it's plausible to me that that again, I'm like, this seems pretty unlikely, but it's plausible to me that something you see, like, big gains once people figure out how to actually train on that. And, like, maybe you you need some amount of kind of expensive to gather training data that people sort of haven't bothered to get yet, but you don't need a huge number of data points because you're not instilling this whole new capability. You're just, like, eliciting. Okay. Actually, use your knowledge of all of the papers you've read in all of these different fields to, like, you know, iterate through these, like, no. These ideas aren't promising. These ones are. Yeah. I again, I think this is 1 of the things that I more think of as this being measured a reasonable amount within you know, just, like, do this 8 hour ML task in a novel domain, you know, with this weird weird constraint or something. Like, it it does seem to me like you have to do some amount of being like, okay. Which things are promising to think about? How would I know if this is, you know, making progress? You know, how should I allocate my time? You know, I've got some limited time resources. How should I allocate my time time to what's most promising? Like, you have to be able be doing some of that. And I think, you know, relative to humans, moles are doing more at you know, more of just like, well, they're quick to implement things or they implement it better or they, you know, they implement more things, and they can then get to to test them or something. But I I think it would be sort of surprising if there's none of that. And if you are I think if you are seeing performance on, you know, long verifiable tasks that are very hard, then in the middle of those tasks, like, where you don't directly have a signal, you you are doing this, you know, nonverifiable task thing of, like, choosing what to spend your time on and and choosing what approach to pursue and deciding whether that was actually working and, you know you know, in terms of you you could sort of put some metric on you know, make 1000000000 dollars or or or something like that. You know, you'd be like, oh, this is actually a verifiable task because there's, like, a number at the end, but that can still involve a whole load of things that look more like what you're describing and less like sort of dumb hill climbing.

Tim Scarfe 01:52:23

Folks, we we I think we've run out of time, but, it's been such an honor to have you both on. May may maybe just in closing, could could you just both say, like, what what is the the single biggest inference that, you know, people out there should be making from the research that you're doing? And and thank you both so much for coming on. It's been an honor.

David Rein 01:52:40

To me, the biggest thing is AI. It might really, you know, totally transform the world, economically and and and socially. And I don't think that, you know, it's certain exactly how that how that'll look, but I think the the rate of progress, speaks to that.

Beth Barnes 01:52:56

It is possible both for things to currently be overhyped and exaggerated and less impressive than they look and for it to be the case that in future, this thing is gonna be a big deal and, like, you should be worried about where that's going. Like, these 2 things can coexist, and I think people often sort of you know, positions are surprisingly correlated on some axis of, like, how, you know, how soon you think AIs or how good you think AIs or something. I'm like, no. These things could all be separate. Like, people can be wrong in different directions simultaneously or whatever.

References

[1] **METR — Model Evaluation and Threat Research**
https://metr.org/
tool METR's home page — the organisation Beth founded and David works with. Useful canonical link for discovery.

[2] **GPQA: A Graduate-Level Google-Proof Q&A Benchmark**
https://arxiv.org/abs/2311.12022
paper David Rein is first author. Tim name-checks GPQA when introducing David, and David cites it as the benchmark that originally got him thinking about scalable oversight.

[3] **Model Evaluation for Extreme Risks**
https://arxiv.org/abs/2305.15324
paper Shevlane et al.'s extreme-risks evaluation framework — the foundational dangerous-capabilities paper behind METR / ARC Evals methodology.

[4] **GSM-Symbolic: Understanding the Limitations of Mathematical Reasoning**
https://arxiv.org/abs/2410.05229
paper

[5] **ARC Prize / ARC-AGI**
https://arcprize.org/
tool Tim repeatedly invokes Chollet's benchmark when discussing the gap between LLM pattern-matching and human intelligence, and the regression-to-the-mean worry about adversarial benchmarks. Chollet also returns in the closing on intelligence-as-a-smooth-ball.

[6] **The Challenge of Rigorously Measuring AI Abilities**
https://melaniemitchell.substack.com/p/the-challenge-of-rigorously-measuring
other

[7] **Measuring AI Ability to Complete Long Tasks**
https://arxiv.org/abs/2503.14499
paper The METR Time Horizons paper — the foundational work the entire episode revolves around, introducing the human-time-to-complete metric and the exponential time-horizon trend.

[8] **METR blog: Measuring AI ability to complete long tasks**
https://metr.org/blog/2025-03-19-measuring-ai-ability-to-complete-long-tasks/
other METR blog post companion to the Time Horizons paper. Use this when linking to the headline graph David and Beth describe.

[9] **Daniel Kokotajlo on 80,000 Hours — AI 2027 updates**

https://80000hours.org/podcast/episodes/daniel-kokotajlo-ai-2027-updates-china-ro
other

[10] **RE-Bench: Evaluating Frontier AI R&D Capabilities**
https://arxiv.org/abs/2411.15114
paper The RE-Bench paper — referenced repeatedly when discussing ML research tasks, iterative solutions, baseliner selection, and the negative correlation between years of experience and performance.

[11] **Evaluating Language-Model Agents on Realistic Autonomous Tasks (ARA)**
https://arxiv.org/abs/2312.11671
paper METR ARA / autonomous tasks paper — the precursor to HCAST that established the long-horizon agent evaluation programme Beth ran at ARC Evals and then at METR.

[12] **METR Autonomy Evaluation Resources**
https://metr.github.io/autonomy-evals-guide/
tool METR's open evaluation guide — includes HCAST and Vivaria, the harness Beth references when describing how the agent scaffolding evolved.

[13] **Will MacAskill on the Sam Harris Podcast**
https://www.samharris.org/podcasts/making-sense-episodes
person

[14] **AI 2027**
https://ai-2027.com/
other Daniel Kokotajlo's AI 2027 scenario which directly cites METR's time horizon work. Beth notes that some AI Futures Project models are more sensitive to the METR number than they should be.

[15] **Nicholas Carlini — personal research page**
https://nicholas.carlini.com/
person

[16] **The Adolescence of Technology**
https://www.darioamodei.com/essay/the-adolescence-of-technology
other

[17] **Many SWE-bench-passing PRs would not be merged into main**
https://metr.org/notes/2026-03-10-many-swe-bench-passing-prs-would-not-be-merged
other METR note that Tim cites directly: roughly half of test-passing SWE-bench agent PRs would not be merged by maintainers, and the merge rate rises more slowly than raw pass rates.

[18] **Measuring the Impact of Early-2025 AI on Experienced Open-Source Developer Productivity**
https://metr.org/blog/2025-07-10-early-2025-ai-experienced-os-dev-study/

other METR developer productivity RCT co-authored by David Rein. The counterintuitive result that experienced OSS devs were slower with AI is the backdrop for Tim's scepticism about the automation narrative.

[19] **Alignment Faking in Large Language Models**
https://arxiv.org/abs/2412.14093
paper Ryan Greenblatt's alignment-faking paper, explicitly cited by Tim when listing the mentalistic vocabulary around scheming.

[20] **Frontier Models Are Capable of In-context Scheming**
https://arxiv.org/abs/2412.04984
paper

[21] **Apollo Research — Scheming Reasoning Evaluations**
https://www.apolloresearch.ai/research/scheming-reasoning-evaluations
other

[22] **Alignment Faking — Anthropic research blog**
https://www.anthropic.com/research/alignment-faking
other

[23] **Emergent Misalignment**
https://arxiv.org/abs/2502.17424
paper

[24] **Role play with large language models**
https://www.nature.com/articles/s41586-023-06647-8
paper

[25] **The Mind is Flat**
https://press.princeton.edu/books/hardcover/9780300238723/the-mind-is-flat
book

[26] **Recent Frontier Models Are Reward Hacking**
https://metr.org/blog/2025-06-05-recent-reward-hacking/
other METR reward-hacking blog that Beth introduces. Includes the finding that models know what is unintended and still do it, plus the RE-Bench vs HCAST asymmetry.

[27] **Early Work on Monitorability Evaluations**
https://metr.org/blog/2026-01-19-early-work-on-monitorability-evaluations/
other METR monitorability work Beth references when discussing chain-of-thought faithfulness, hidden computation, and the distinction between faithfulness and monitorability.

[28] **Chain of Thoughtlessness? Analysis of CoT in Planning**
https://arxiv.org/abs/2405.04776

paper