MY WEIRD PROMPTS

Podcast Transcript

EPISODE #67

AI & Code: Scaling or Pivoting?

Published December 21, 2025 • Runtime: 22:30

https://myweirdprompts.com/episode/20251221-162001/

EPISODE SYNOPSIS

Join Corn and Herman on "My Weird Prompts" as they tackle Daniel Rosehill's burning question: are large language models the right tool for writing computer code? They dissect whether simply scaling up current LLMs will fix their coding flaws or if a fundamental architectural pivot is needed. From the messy nature of human language versus the binary logic of code, to the concept of "Verifiable AI" and the emergence of "Large Reasoning Models," this episode explores the future of AI in programming by 2026, offering insights for both skeptical users and tech enthusiasts alike.

DANIEL'S PROMPT

Daniel

I'd love to turn it over to you and get your opinion as to whether you think large language models are inherently suitable for this task in the first place. If you think about them as being language models, that almost infers that they're for generating human language, or maybe that was what they were expected to do, and code is very, very different. As AI evolves, there are two potential paths to getting code generation AI tools from where we are now to where we could be. The first is hoping on scale that we're going to scale up compute, scale up the models, scale up the context window. Everything is going to get bigger and better until these challenges just aren't in the way to any significant extent. In other words, we're going to scale and engineer our way past this blocker. The second, which I don't even have any thoughts as to what this might look like, but that's where I'd love to hear your thoughts, is it going to be a more fundamental pivot towards a different type of AI model entirely than the LLM? Maybe we'll see a bifurcation between LLMs doing conversational tasks and code generation models doing something very different. Where do you see it moving as we look towards 2026?

TRANSCRIPT

Corn

Welcome to another episode of My Weird Prompts! I am Corn, your resident sloth who likes to take things slow and really chew on a topic, and I am joined as always by my much speedier, much more opinionated friend, Herman. Today we are diving into a really juicy prompt from the show's producer, Daniel Rosehill. He sent us a question that cuts right to the heart of the tech world right now. Are large language models actually the right tool for writing computer code?

Herman

Hello, everyone. I am Herman Poppleberry, and yes, I have my donkey ears perked up for this one because it touches on a fundamental misunderstanding of what these models actually are. Daniel is asking whether we are just scaling our way toward a wall or if we need a total pivot in how AI handles programming. It is a question of architecture versus brute force.

Corn

Right, because when you think about it, language is messy. It is full of metaphors, slang, and vibes. But code? Code is logic. It is binary. If you miss a semicolon, the whole thing crashes. So, why are we using a language model to do a logic job? Does that even make sense, Herman?

Herman

Well, Corn, I would start by pushing back on the idea that language and code are as different as you think. From a mathematical perspective, both are sequences of tokens with underlying structural rules. However, you are right that the stakes are different. If I tell you a story and get a word wrong, you still get the gist. If an AI writes a Python script and hallucinates a library that does not exist, the script is useless.

Corn

Exactly! And that is the core of the prompt today. Are we going to see these models just get bigger and bigger until they stop making those silly mistakes, or are we going to see a split? You know, like a brain where one side does the talking and the other side does the math?

Herman

That is the bifurcation theory Daniel mentioned. But before we get to 2026, we have to look at where we are. Right now, we are in the era of scale. We are throwing more GPUs and more data at the problem. We have models with context windows of over a million tokens now. But is a bigger bucket really the answer if the water inside is still a bit murky?

Corn

I mean, I like a big bucket. It lets me remember what I said ten minutes ago, which, as a sloth, is a real luxury. But I see your point. If the model is just predicting the next most likely character based on what it saw on GitHub, it is not really thinking like a programmer, is it? It is just a very fancy autocomplete.

Herman

I think calling it fancy autocomplete is a bit of an oversimplification, Corn. It is more like a statistical intuition of logic. But here is where I might disagree with the premise that we need a completely different model. There is a concept called the Scaling Laws. Many researchers believe that as you increase compute and data, emergent properties appear. Things like reasoning and logic seem to emerge from the language processing.

Corn

But do they really emerge, or is it just a very good imitation? I feel like I have seen these models get stuck in loops or fail at basic math even when they can write a beautiful poem. If we are looking toward 2026, can we really expect a language-based system to suddenly understand the deep architectural requirements of a complex software system?

Herman

That is the trillion-dollar question. I would argue that we are already seeing the beginning of the pivot Daniel mentioned, but it is not a pivot away from language models. It is an augmentation of them. We are seeing things like Tree of Thoughts or chain of thought reasoning where the model is forced to check its own work.

Corn

Okay, but isn't that just adding more layers of language on top of language? It is like me asking myself if I am sure I want to eat this hibiscus flower, and then answering myself, and then checking that answer. It still does not change the fact that I am a sloth who just wants flowers. If the model is fundamentally a word-predictor, no amount of self-correction changes its DNA.

Herman

See, that is where you and I differ. I think the DNA of an LLM is more flexible than you give it credit for. However, I will concede that for high-stakes enterprise coding, we might need what is called a symbolic AI hybrid. This would be a system where the LLM dreams up the code, but a rigid, logical engine—a symbolic model—verifies it against the laws of mathematics and syntax before a human ever sees it.

Corn

Now that sounds like a plan. It is like having a creative writer and a grumpy editor working together. Speaking of people who have a lot to say, let's take a quick break for our sponsors. Larry: Are you tired of your shoes just sitting there, being shoes? Do you wish your footwear had more... ambition? Introducing the Gravity-Go Boots! These are not just boots; they are a lifestyle choice. Using patented Unobtainium-mesh technology, Gravity-Go Boots make you feel lighter than air, mostly because the soles are filled with a proprietary pressurized gas that we definitely checked for safety. Walk on water! Walk on walls! Walk on your neighbor's roof! Note: Gravity-Go Boots may cause unexpected floating, temporary loss of bone density, or a sudden, uncontrollable urge to migrate south for the winter. No returns, no refunds, no regrets. Gravity-Go Boots. BUY NOW!

Corn

...Thanks, Larry. I think I will stick to my natural claws for climbing, personally. Anyway, back to the future of AI and code. Herman, you mentioned this hybrid idea. If we look at the timeline toward 2026, do you think we will actually see specialized coding models that don't speak English at all?

Herman

I actually think that would be a step backward. The power of current AI tools like GitHub Copilot or Cursor is that they understand the intent. You can talk to them. If you have a model that only understands the logic of C-plus-plus but cannot understand a human explaining a business problem, you have just built a very expensive compiler. The magic is in the translation from human messiness to machine precision.

Corn

I see that, but I wonder if we are hitting a plateau. We have scraped most of the public code on the internet. Where does the new data come from? If we just keep training on AI-generated code, don't we get a sort of digital inbreeding? The errors just get baked in deeper.

Herman

That is a very astute point, Corn. It is called model collapse. To get to the 2026 goals Daniel is talking about, we have to move beyond just scraping the internet. We need synthetic data. We need models that can play against themselves, sort of like how AlphaGo learned to beat the world champion at Go. It didn't just read books about Go; it played millions of games against itself to find new strategies.

Corn

So, you're saying the AI should write code, run it, see it fail, and then learn from that failure? Like a little digital laboratory?

Herman

Exactly. That moves it away from being just a language model and into being an agent. An agent can interact with a terminal, run a test suite, and iterate. By 2026, I suspect we won't be talking about LLMs for code. We will be talking about Large Reasoning Models or L-R-Ms. The language will just be the interface, not the engine.

Corn

I like the sound of that. It feels more robust. But wait, if it gets that good, does it mean humans stop learning how to code? Because if the AI is doing the thinking and the checking and the iterating, I am just the guy sitting there saying, make me a website that sells hats for sloths.

Herman

And that is exactly why some people are skeptical. They think we are losing the fundamental skill of logic. But look, we said the same thing about calculators and math, or compilers and assembly language. Each layer of abstraction allows us to build bigger things. The coder of 2026 won't be someone who worries about syntax; they will be a systems architect.

Corn

I don't know, Herman. There is something about knowing how the gears turn. If I don't know how to climb the tree myself, I am in trouble when the elevator breaks. I think there is a real risk in moving too fast toward total automation.

Herman

It is not about total automation; it is about moving the human to a higher level of oversight. But I can tell you are unconvinced. Let's see if our caller has a more grounded perspective. We have Jim from Ohio on the line. Jim, what do you think about AI writing code? Jim: Yeah, this is Jim from Ohio. I've been listening to you two yappers for ten minutes and I still don't know what a token is and I don't care to know. You're talking about 2026? I'm worried about 2024! My neighbor, Miller, bought one of those smart lawnmowers and it ended up in the creek three times last week. The thing has a brain the size of a pea and he thinks it's the future.

Corn

Well, Jim, that's a fair point about real-world reliability. But what about the software side? Jim: It's all junk! I tried to use one of those chatbots to help me write a simple script for my spreadsheet—just something to track my collection of vintage hubcaps—and the thing gave me code that looked like Greek. Didn't work. Kept telling me it was sorry. I don't need an apology from my computer; I need it to work! And don't get me started on the weather. It's been raining for three days and my cat, Whiskers, is acting like it's the end of the world. Just pacing back and forth. You guys are talking about computers thinking? My cat can't even figure out that the rain is outside, not inside!

Herman

Jim, I think your experience with the spreadsheet script is actually what we are talking about. Current models often fail at those specific, logical tasks because they are prioritizing looking right over being right. Jim: Exactly! It's all show and no go. Back in my day, if you wanted a program, you sat down and you wrote it. You didn't ask a magic box to guess what you wanted. You're all just getting lazy. And by the way, the coffee at the diner this morning was lukewarm. If they can't even get a pot of coffee right, how are they going to get a computer to write code? It's all going to pot.

Corn

Thanks for the call, Jim. Always good to have a reality check from Ohio.

Herman

Jim is the perfect example of the user who will break the 2026 models. He doesn't want to prompt-engineer; he wants results. And that brings us back to Daniel's question about the path forward. Is it scale or a pivot? I think the pivot is toward what I call Verifiable AI.

Corn

Verifiable AI? Explain that to me like I am... well, a sloth.

Herman

It means that before the AI gives you the code, it has to prove it works. It runs it in a sandboxed environment, checks the output, and if it fails, it loops back and fixes it without ever bothering you. The user only sees the final, working product. That requires a fundamental shift from a one-shot generation model to an iterative agentic model.

Corn

So, it's not a different model entirely, but a different way of using the model?

Herman

It is both. You need the model to be trained specifically on execution traces—seeing how code runs—rather than just the text of the code itself. Most LLMs today have never seen code actually execute. They have only seen the static text on a page. That is like trying to learn to drive by reading a car's manual but never actually sitting in the driver's seat.

Corn

That is a great analogy. If I read a book about climbing, I might know where my hands go, but until I feel the bark under my claws, I don't really know how to climb. So, for the 2026 horizon, are we looking at models that have essentially spent millions of hours in a virtual simulator?

Herman

Precisely. We are seeing this with companies like OpenAI and Anthropic already. They are starting to integrate tools directly into the model's thought process. By 2026, the distinction between a language model and a coding tool will be much sharper. We might have a general-purpose brain that calls upon a specialized coding lobe when it detects a programming task.

Corn

I can see that. But I want to go back to Daniel's point about the context window. He mentioned scaling up everything—compute, models, context. If I can fit an entire library of code into the context window, doesn't that solve the problem of the AI not knowing how the whole system works?

Herman

Not necessarily. Just because you can read a thousand books doesn't mean you can synthesize them into a coherent plan. A massive context window is great, but it increases the noise. The model can get lost in the details and forget the main objective. Scaling the window is a brute-force solution to a structural problem.

Corn

So, you're on team pivot?

Herman

I am on team evolutionary pivot. I don't think we throw away the transformers or the large language models. They are too good at understanding us. But we have to wrap them in a layer of formal logic. We need a system where the AI says, I think this is the code, and then a separate, non-AI system says, Let me check the math on that.

Corn

See, I think I am more on the side of specialization. I think we might see a world where we have models that don't speak a word of English. They just speak Python. And you have an LLM that acts as the translator. It seems more efficient to me. Why waste all that brainpower on Shakespeare when you are just trying to optimize a database?

Herman

Because the database optimization depends on the context of the business! If the AI doesn't understand that the database is for a hospital and must prioritize data privacy and speed of access for emergencies, it might optimize for the wrong thing. You cannot decouple the human context from the technical execution. That is why the language aspect is so vital.

Corn

Hmm. You've got me there. A purely logical model might find the most efficient solution that is also completely unethical or useless for a human. It's like a genie that gives you exactly what you asked for, but not what you wanted.

Herman

Exactly. The language model is the soul of the machine; the logical engine is the hands. You need both. Looking at the 2026 target, I think we will see the rise of small, highly specialized models that are incredibly good at logic, which are then orchestrated by a central, highly capable LLM.

Corn

So, like a conductor and an orchestra. The conductor knows the music and the vibe, and the violinists just know how to play the violin perfectly.

Herman

That is a beautiful way to put it, Corn. Though I suspect the violinists in this case are actually just very fast calculators.

Corn

Hey, don't knock the violinists! So, let's talk practicalities. If I'm someone looking at this from the outside, what do I do with this information? Does it mean I should stop learning to code?

Herman

Absolutely not. It means you should change how you learn. Stop memorizing syntax. Stop worrying about where the brackets go. Focus on problem decomposition. Learn how to break a big, messy human problem into small, logical steps. If you can do that, you can lead an AI to build anything. If you can't do that, the AI will just build you a very fast version of a mistake.

Corn

I love that. Focus on the what and the why, and let the machine handle the how. But you still need to know enough of the how to know when the machine is lying to you.

Herman

Correct. You have to be the supervisor. You have to be the one who looks at the AI's work and says, This looks efficient, but it's going to be a nightmare to maintain in six months. That kind of foresight is still uniquely human—or donkey, as the case may be.

Corn

Or sloth! Don't forget us slow thinkers. We have the best foresight because we have plenty of time to look ahead while we're moving.

Herman

Fair enough. So, to wrap up Daniel's prompt, it seems we're looking at a future that isn't just about making the same models bigger. It's about making them smarter by giving them tools, feedback loops, and a bit of a logical backbone.

Corn

It's a transition from a talking head to a working hand. I think it's exciting, even if it's a bit scary. I just hope the AI doesn't start asking for its own coffee. Jim from Ohio would have a heart attack.

Herman

I think Jim would just complain that the AI's coffee is too hot. But in all seriousness, the next two years are going to be a wild ride in software development. We are moving from the era of writing code to the era of steering code.

Corn

Well, I for one am ready to steer. As long as the steering wheel is made of something soft and I can do it from a hammock.

Herman

I wouldn't expect anything less.

Corn

This has been a fascinating look at the future of AI. Thank you to Daniel Rosehill for sending in such a brain-bending prompt. It really made us—well, mostly Herman—think.

Herman

I think you contributed some very important sloth-like wisdom, Corn. The idea of not losing our own skills while the machines get better is a vital takeaway.

Corn

Thanks, Herman. And thank you all for listening to My Weird Prompts. You can find us on Spotify and all the other places you get your podcasts. We'll be back next time with more strange ideas and hopefully fewer floating boots.

Herman

Yes, please avoid the floating boots. Until next time, keep your logic sharp and your prompts weirder.

Corn

See ya