Podcast Transcript

EPISODE #52

System Prompts vs. Fine-Tuning: Are We Building Solutions for Problems That Don't Exist?

Published December 11, 2025 • Runtime: 29:45

https://myweirdprompts.com/episode/system-prompts-vs-fine-tuning-are-we-building-solutions-for-/

EPISODE SYNOPSIS

Is all the infrastructure around fine-tuning actually solving real problems, or are we chasing solutions looking for problems? In this episode, Corn and Herman dive deep into Daniel's question about system prompting versus fine-tuning in AI systems. They explore how system prompts actually work, why they're surprisingly effective, and whether the massive investment in fine-tuning platforms matches the real-world demand. Plus, they discuss how new tools like the Model Context Protocol might be changing the game entirely—and whether most companies even need to fine-tune at all.

DANIEL'S PROMPT

Daniel

Hello there, Herman and Cord. I very much enjoying listening to your responses to my question earlier today about multi-agentic frameworks and the potential utility in exploring policy. And you actually in your answer described it better than I did in my prompt or what I was trying to get to, which was you described it as stress testing policy proposals. That's exactly what I was asking about specifically, and seeing how multi-agentic frameworks might be useful in that. Now, in the course of your answer, you explained system prompting, which was interesting because system prompts are kind of how I became really interested in AI and just enraptured, I think, by the ability to use natural language rather than code to create very powerful systems potentially that really could steer a model in a very specific direction. And over time, we've seen AI tools become, agentic AI become more sophisticated and a lot of the focus at the moment is on MCP and, you know, this kind of question of how can we get AI agents to actually do things in a way that is safe. But I actually think that system prompts are one of the most useful parts of the AI picture and even in agentic systems, I've come to see that they actually maintain a role. I've always kind of wondered what's going to happen to assistance now that everything's becoming about agents. But even when you're dealing with real agentic frameworks where you have maybe some degree of autonomy, it's still useful to have a system prompt because that can actually really change the course of a model's behavior. Another thing we've talked about lately a lot is fine-tuning, and my first experiences in fine-tuning came in the ASR domain, fine-tuning an ASR model. But something that Herman mentioned, which got me thinking is, you know, he explained that system prompting is almost like telling an actor to behave in a certain way and that creates a constraint through which they kind of respond. And in my experience, it's actually pretty much the case. I've played around with different system prompts. Some kind of just for fun, but some to actually see how far can you go with system prompting? And I've tried kind of permutations like, "You must respond with the minimum where it's possible," trying to get around kind of the kind of verbosity you see in AI systems, and just respond "yes" or "no." And it will actually, the model will adhere to that. Now here's my question. System prompting, writing a system prompt, is obviously a much simpler undertaking, technically and from a time perspective, than even fine-tuning. So, but at the same time, it's kind of hard to believe that simply writing an instruction is going to deeply change the model. And of course, it's not the weights of the model are remaining the same. But I'd love to hear you guys, hear your thoughts about the role of system prompting and that making sense as a kind of very fairly simple and accessible way of changing AI experiences and when that's not going to cut it and fine-tuning might be more effective. And continuing with that thought, fine-tuning large language models, there's an increasing amount of infrastructure supporting fine-tuning, and I'd love to hear about, you know, I sometimes see questions on YouTube and Reddit like, "Has anyone actually found a good reason for fine-tuning?" And there seems to be a lot of infrastructure for it, and I'm not quite sure myself in practice who's benefiting from fine-tuning, but I'll turn it over to you now for the answers.

TRANSCRIPT

Corn

Welcome back to My Weird Prompts, the podcast where producer Daniel Rosehill sends us fascinating questions and we dive deep into them with absolutely no preparation whatsoever. I'm Corn, and I'm joined as always by Herman Poppleberry. Today we're tackling something that's been on Daniel's mind lately - the role of system prompting in AI, how it compares to fine-tuning, and whether all this infrastructure we're building around fine-tuning is actually solving real problems or just creating solutions looking for problems.

Herman

Yeah, and what I find particularly interesting about this prompt is that it gets at something fundamental about how we interact with AI systems that I think a lot of people overlook. Everyone's focused on agents and MCP and autonomous systems, but the unglamorous work of actually steering model behavior? That's often happening in a system prompt, and it's kind of invisible.

Corn

Right, exactly. Like, Daniel mentioned that he got really enraptured by the idea that you could use natural language instead of code to create powerful systems. And I think a lot of people don't realize that's... kind of wild when you think about it. You're literally just writing instructions in English and the model goes, "Okay, I'll do that."

Herman

Well, hold on. I want to be careful about how we frame this, because saying "you're just writing instructions" kind of undersells what's actually happening. It's not magic - there are real mechanisms at play. The system prompt is being processed by the model in a specific way, and it does genuinely constrain behavior, but it's not like you're rewriting the weights or fundamentally changing the model's architecture.

Corn

Okay, but like... it kind of does work though. Daniel mentioned he tried system prompts that said things like "respond with the minimum possible" or "only answer yes or no," and the model actually adheres to that. That's not nothing.

Herman

It's not nothing, but I'd push back on the framing that it's simple. What's happening is that you're providing context that the model uses to make predictions about what tokens should come next. The model has been trained on examples of concise responses, verbose responses, yes-or-no answers, all of it. So when you add a system prompt that says "be concise," you're essentially biasing the probability distribution of what comes next toward shorter, more direct tokens. It works because the capability is already there in the model.

Corn

Okay, so it's like... the model already knows how to be concise, and the system prompt is just saying, "Hey, use that capability now."

Herman

Exactly. And that's actually why system prompting is so powerful but also why it has limits. You can't use a system prompt to make a model do something it was never trained to do. If the base model doesn't have the capability, no amount of clever prompting will create it from nothing.

Corn

Right, which makes sense. So if a model has never learned to write in Klingon or calculate in base-7 or whatever, you can't just prompt it into doing that. But here's what I'm curious about - if that's true, then why is fine-tuning even necessary? Like, couldn't you just... train the model better in the first place?

Herman

Well, that's a different question. Fine-tuning is about specialization. You take a general-purpose model and you adapt it to a specific domain or task. The infrastructure around fine-tuning has exploded in 2024 - there are all these platforms now that make it accessible. But here's the thing that I think Daniel's getting at in this prompt: we're not entirely sure who's actually benefiting from it in practice.

Corn

Yeah, that was interesting. Daniel said he sees questions on Reddit and YouTube like, "Has anyone actually found a good reason to fine-tune?" And that's... a pretty damning question for all this infrastructure that's been built, right?

Herman

It is, but I think the answer is more nuanced than it might seem. Fine-tuning is genuinely useful in certain domains. Medical specialties, for instance - there's research showing that fine-tuning LLMs for specific medical subspecialties improves accuracy significantly. Sentiment analysis, named entity recognition, specialized summarization tasks - these all benefit from fine-tuning. But you're right that there's this weird gap between the hype around fine-tuning and the actual adoption.

Corn

So what's causing that gap? Is it that the use cases are niche?

Herman

Partially. But I think it's also a matter of cost-benefit analysis. Fine-tuning requires you to have good training data, which is hard to come by. It requires infrastructure and expertise. It's more complex than prompt engineering. And here's the thing - in a lot of cases, a really well-crafted system prompt gets you 80% of the way there for a fraction of the effort.

Corn

But isn't that exactly what Daniel's asking though? Like, how do we know when we've hit the ceiling of what system prompting can do? How do you know when it's time to actually fine-tune?

Herman

That's the million-dollar question, honestly. And I don't think there's a clean answer. But we can talk about the trade-offs. System prompting is fast, cheap, and reversible. You can iterate rapidly. Fine-tuning is slower, more expensive, and requires commitment. But fine-tuning can achieve higher accuracy on specialized tasks because you're actually modifying the model's learned behaviors through additional training on domain-specific data.

Corn

Let's take a quick break from our sponsors. Larry: Tired of your AI system prompts being boring and ineffective? Introducing PromptMeld 3000 - the revolutionary prompt crystallizer that uses proprietary harmonic resonance technology to amplify your system prompts to their maximum potential. Simply speak your desired AI behavior into the PromptMeld 3000, and it will somehow... enhance it. Users report that their AI responses feel "more AI-like" and their productivity has increased by approximately "some amount." PromptMeld 3000 - because if you're going to overthink your prompts, you might as well have a device. BUY NOW!

Herman

...Right. Thanks, Larry. Anyway, where were we? Oh yes, the actual mechanics of this.

Corn

So let me try to ground this in something concrete. Daniel mentioned he had his first experiences with fine-tuning in the ASR domain - that's automatic speech recognition, right? What was he doing there?

Herman

ASR is actually a pretty good example because it's a domain where fine-tuning has proven its worth. If you want a speech recognition model to understand a specific accent, industry jargon, or acoustic environment, fine-tuning on domain-specific audio data genuinely improves performance. You can't just prompt your way to understanding medical terminology in a thick Scottish accent - the model needs to have learned that pattern.

Corn

Okay, so that makes sense. But then why does that not translate to language models? Like, why can't we just fine-tune every LLM for every specialized domain?

Herman

We could, but the economics don't work. Speech recognition has a clear value proposition - if your transcription accuracy goes from 85% to 95%, that's a huge win. Language models are more... fuzzy. It's harder to measure whether you've actually improved performance, and the cost of fine-tuning has to be justified against the benefit. For a lot of use cases, prompt engineering and system prompts get you good enough results.

Corn

But "good enough" is different from "optimal," right? I mean, if you're a financial services company and you need consistent, specialized behavior from an AI system, wouldn't you want to fine-tune?

Herman

You might, and some do. But here's where I think the infrastructure question comes in. There's been this massive investment in fine-tuning platforms and services. OpenAI has fine-tuning capabilities, Anthropic has them, there are startups building entire businesses around it. But the demand hasn't quite materialized the way the supply suggests it should. That's what Daniel's picking up on.

Corn

So is it a chicken-and-egg thing? Like, did we build the infrastructure before we understood the actual use cases?

Herman

I think there's some of that, yeah. But I also think there's something else happening. The industry recognized that fine-tuning would be important, so they built tools for it. But they didn't necessarily anticipate how good system prompting would get, or how much you could accomplish with clever prompt engineering and the Model Context Protocol.

Corn

Oh, that's a good point. Daniel mentioned MCP in the prompt - the Model Context Protocol. Anthropic released that in November 2024, right?

Herman

Right, and this is actually crucial context for this conversation. MCP is changing how agents interact with tools and context. It streamlines agent interactions and reduces context usage. And here's the thing - MCP actually might be a partial answer to some of the problems that people thought fine-tuning would solve. If you can give an agent access to the right tools and context through MCP, you might not need to fine-tune the base model at all.

Corn

So you're saying MCP could be a substitute for fine-tuning in some cases?

Herman

In some cases, potentially. Not all. But the point is that the landscape is shifting faster than people anticipated. We have better ways to constrain and direct model behavior now - through system prompting, through tools and context, through MCP. That changes the calculus around fine-tuning.

Corn

But doesn't that mean we should be having this conversation about what fine-tuning is actually for? Like, let's define the actual use cases where it makes sense.

Herman

Absolutely. And I think there are real ones. Medical specialization is one. Financial modeling and analysis - there are companies that have built proprietary datasets and fine-tuned models for specific financial tasks. Legal document analysis in specialized areas of law. These are domains where you have: One, high-value tasks where accuracy really matters. Two, domain-specific data that you own or can access. Three, a clear performance metric you're trying to optimize.

Corn

Okay, so those are the criteria. But most companies probably don't have all three of those things, right?

Herman

No, most don't. Most companies have maybe one or two. And for those cases, system prompting plus good tool integration through something like MCP is probably sufficient.

Corn

So circling back to Daniel's original question - why is system prompting so effective even though it seems so simple? You're not changing the weights, you're not retraining anything. You're just writing instructions.

Herman

Right, and I think the answer is that the model has been trained on vast amounts of text that includes examples of following instructions. The system prompt is just one more piece of context that the model processes. It's not magic, but it's also not trivial. The model has learned to be sensitive to instructions, to adapt its behavior based on framing and context.

Corn

But here's what I don't quite get - and I think this is where Daniel's skepticism is coming from - how does that work mechanistically? Like, how does writing "respond concisely" in a system prompt actually change the probability distribution of the tokens that come out?

Herman

That's a great question, and honestly, we don't have perfect visibility into this. The models are trained end-to-end, and the system prompt is just part of the input. During training, the model learned patterns about what kinds of responses follow what kinds of prompts. When you add a system prompt at inference time, you're essentially activating those learned patterns. It's like... imagine an actor who has learned to play both comedic and dramatic roles. If you tell them, "This is a comedic scene," they'll automatically adjust their performance because they've learned the patterns of comedy. The system prompt is that direction.

Corn

Okay, I like that analogy. But here's where I push back a little - Daniel mentioned he's tried to get the model to respond with just "yes" or "no," and it actually works. But that seems like a pretty strong constraint. How does the model learn to be that constrained?

Herman

Because it has seen examples of yes-or-no responses in its training data. It understands that constraint. And here's the thing - that's actually a really good test case for the limits of system prompting. If you ask a model to respond in a way it has genuinely never seen in training data, the system prompt won't work. But yes-or-no responses? That's a pattern the model has seen millions of times.

Corn

So system prompting works because it's activating existing capabilities, but fine-tuning is about creating new capabilities?

Herman

I'd say it's more about strengthening and specializing existing capabilities. Fine-tuning isn't creating something from nothing either. You're taking a pretrained model and showing it more examples of the specific kind of behavior you want. You're essentially saying, "Here are a thousand examples of how to analyze medical documents. Learn from these." The model updates its weights slightly to be better at that specific task.

Corn

And that updates the weights in a way that a system prompt can't?

Herman

Exactly. A system prompt operates at inference time. Fine-tuning modifies the model itself. So if you fine-tune a model on medical documents, it becomes fundamentally better at analyzing medical documents, even without a system prompt telling it to do so. That's a permanent change to the model.

Corn

But that also means you have to deploy a different model, right? You can't just swap it out in your system prompt like you can with instructions.

Herman

Right, and that's part of why system prompting is so appealing. It's reversible, it's fast, it doesn't require redeployment. But that also means it's limited to activating capabilities that are already there.

Corn

Okay, so here's a practical question. If I'm building an AI system right now, in 2024, and I want specialized behavior - let's say I want an AI assistant for a specific industry - when do I start with system prompting and when do I go straight to fine-tuning?

Herman

I'd always start with system prompting. It's faster, cheaper, and you learn a lot about what you actually need. If you can get 90% of the way there with a really well-crafted system prompt, why spend weeks and money on fine-tuning? But if you hit a wall - if you're consistently getting results that aren't good enough - then you evaluate fine-tuning.

Corn

But how do you know you've hit the wall? How do you know you're not just not being creative enough with your system prompt?

Herman

That's fair. You could be. But there are some signals. If you're trying to get the model to understand very specialized terminology or concepts that don't appear much in the training data, system prompting probably won't cut it. If you're trying to get consistent, reproducible behavior on a narrow task where accuracy matters a lot, fine-tuning might be worth it. If you're trying to change the fundamental style or approach of the model - like, make it always reason step-by-step in a specific way - fine-tuning could help. But honestly, a lot of that can be accomplished with really good system prompting too.

Corn

So you're saying the line between them is blurry?

Herman

Very blurry. And I think that's actually the honest answer to Daniel's question about why there's all this infrastructure for fine-tuning but people aren't sure if they need it. The infrastructure got built because the industry thought fine-tuning would be a big deal. But in practice, system prompting has turned out to be more powerful and more flexible than people anticipated.

Corn

But that doesn't mean fine-tuning is useless, right?

Herman

No, not at all. It's useful. It's just more niche than the hype suggested. And here's what's interesting - and I think this is something Daniel's picking up on - we might be entering a phase where the real innovation is in how you combine system prompting, tool integration through MCP, and selective fine-tuning. Not one or the other, but all of them together.

Corn

So like, you have a base model, you give it the right system prompt, you integrate it with tools through MCP, and then maybe you fine-tune on some specific domain data?

Herman

Potentially, yeah. Though I'd say most people would get great results with the first two. The fine-tuning is the cherry on top for high-value, specialized tasks.

Corn

Alright, we've got a caller on the line. Go ahead, you're on the air. Jim: Yeah, this is Jim from Ohio. I've been listening to you two dance around this for the last fifteen minutes and I gotta tell you, you're overcomplicating it. And you know, my neighbor Randy does the same thing - he spent three hours trying to figure out which air filter to buy when he could've just grabbed any of them. Anyway, here's my point. System prompts work because they work. Fine-tuning is just for people who like spending money on complicated solutions. You don't need either one if you just ask the question right the first time.

Corn

Well, I appreciate the feedback, Jim, but I think there's actually more going on under the hood than just "asking the question right." Jim: Yeah, but see, that's exactly the problem. You're making it complicated. In my day, you just told the computer what to do and it did it. Now you've got all these fancy terms and frameworks and... I don't know, it just feels like people are making problems so they can sell solutions. Also, my blood pressure has been acting up lately, so I'm already irritable. But seriously, why can't people just keep things simple?

Herman

I understand the skepticism, Jim, but the thing is, these models are fundamentally different from the computers you're used to. They're probabilistic. They don't just follow instructions - they generate text based on learned patterns. System prompting works, but understanding why it works and when it doesn't is actually important if you want to use these systems effectively. Jim: Ehh, you're still overcomplicating it. My cat Whiskers understands instructions just fine without all this theoretical nonsense. She knows to get off the kitchen counter when I tell her to. Why can't your AI models be that straightforward?

Corn

Well, because cats have learned behavior through experience and punishment, and LLMs have learned through statistical patterns in text. It's actually a pretty different mechanism. Jim: Look, all I'm saying is that you two sound like you're trying to justify why fine-tuning exists when maybe it just doesn't need to. Keep it simple. Thanks for having me on, but I'm not convinced you've made your case. I'm going back to my coffee.

Corn

Thanks for calling in, Jim. We appreciate the perspective.

Herman

Yeah, and look - Jim's not entirely wrong. There is a tendency in the industry to overcomplicate things. But I'd argue that understanding these mechanisms is what prevents overcomplification in practice.

Corn

That's a fair point. So let's talk about what someone should actually do if they're building a system right now. What's the practical takeaway?

Herman

Okay, so here's my recommendation. Start with system prompting. Invest time in writing good prompts. Test them. Iterate. See how far you can get. If you're building something for a specialized domain, layer in tool integration - MCP is good for this. Give your system access to the right context and capabilities. Then, and only then, if you're still not getting the results you need, consider fine-tuning. But be honest about whether you're hitting a real limitation or just need to be more creative with your system prompt.

Corn

And what's the timeline for that? Like, how long should you spend on system prompting before you give up and fine-tune?

Herman

Depends on the task, but I'd say weeks, not months. If you're spending more than a couple of weeks iterating on system prompts without progress, then maybe fine-tuning is worth exploring. But most people don't spend nearly enough time on the prompt engineering phase before they jump to fine-tuning.

Corn

And what about cost? That's a real factor for a lot of companies.

Herman

System prompting is basically free - it's just your time. Fine-tuning has real costs. You need training data, you need compute, you need infrastructure. It's not huge, but it's not nothing. So yeah, cost is definitely a reason to exhaust the system prompting avenue first.

Corn

What about the Model Context Protocol? Should people be thinking about that now?

Herman

Absolutely. MCP was released in November 2024, and it's becoming the standard way for agents to interact with tools and context. If you're building agentic systems, you should be thinking about MCP. It actually changes what's possible with system prompting and tool integration, because it makes it easier and more efficient to give your agent access to the right context and capabilities.

Corn

So MCP is kind of the glue that ties system prompting and tool integration together?

Herman

Exactly. It's a protocol that standardizes how agents interact with their environment. Code execution with MCP is more efficient, uses fewer tokens, reduces context. That's huge for building effective systems.

Corn

Alright, so the takeaway is: system prompting is more powerful than people realize, fine-tuning is more specialized than the hype suggests, and MCP is changing the game for how agents interact with their environment.

Herman

I'd add one more thing. This stuff is evolving really fast. Six months from now, the landscape might look different. We might have better tools, better models, better ways to constrain and direct behavior. So the advice I'm giving today should be taken with that in mind.

Corn

That's a good caveat. And I think that's actually part of what Daniel was getting at in this prompt - there's a lot of infrastructure and hype around fine-tuning, but the fundamentals of what works and what doesn't are still being figured out.

Herman

Right. And I think as the industry matures, we'll get clearer answers. We'll see which fine-tuning use cases actually pan out, which ones were hype, and what new approaches emerge. But right now, we're in a phase where system prompting is doing a lot of heavy lifting.

Corn

So if someone's listening to this and they're curious about experimenting with system prompting, what should they do?

Herman

Pick a task you care about. Write a detailed system prompt that explains exactly how you want the model to behave. Test it. Try variations. See what works. Don't assume you need fine-tuning until you've really explored what's possible with system prompting. And if you do find yourself wanting to fine-tune, be honest about why. Is it because you've genuinely hit a limitation, or is it because you heard it's the cool thing to do?

Corn

Yeah, and I think that honesty is important because there's definitely some cargo cult stuff happening in AI right now. Like, people see that fine-tuning exists and assume they need it, without really thinking about whether it solves their problem.

Herman

Exactly. And that's why understanding the difference between system prompting and fine-tuning, and when each is appropriate, is actually really valuable. It prevents you from wasting time and money on solutions you don't need.

Corn

Alright, so as we wrap up, I think the key insight here is that system prompting is underrated and fine-tuning is maybe overhyped. But both have their place, and the smart approach is to start simple and add complexity only when you need it.

Herman

I'd agree with that. And I'd add that the whole landscape is changing with things like MCP. The tools we have now for controlling and directing AI behavior are getting better and more sophisticated. So even if you think you need fine-tuning today, you might not need it six months from now because there's a better way to do what you're trying to do.

Corn

That's a really good point. Thanks to Daniel Rosehill for sending in this prompt. It's exactly the kind of question that gets at the real substance of how these systems work, beyond the hype and the marketing.

Herman

Yeah, and I think it's a question more people should be asking. Not just "Can I fine-tune?" but "Do I actually need to?" There's a lot of wisdom in that skepticism.

Corn

Alright, everyone, thanks for listening to My Weird Prompts. You can find us on Spotify and wherever you get your podcasts. If you've got a weird question or prompt you think we should explore, send it our way. And if you're interested in system prompting, fine-tuning, or the Model Context Protocol, there's never been a better time to experiment. The tools are out there, they're accessible, and the only thing stopping you is your own curiosity.

Herman

And maybe a well-crafted system prompt.

Corn

Yeah, that too. Thanks for listening, everyone. We'll catch you next time on My Weird Prompts.