# SDS PODCAST EPISODE 959: BUILDING AGENTS 101: DESIGN PATTERNS, EVALS AND OPTIMIZATION (WITH SINAN OZDEMIR)

**Show Notes:** http://www.superdatascience.com/959

| | | |
|---|---|---|
| Jon Krohn: | 00:00:00 | Want to be building and deploying Agentech AI, but don't know where to start. Today's episode will provide you with an introductory 101 type course on everything you need to know. |
| | 00:00:10 | Welcome to episode number 959 of the SuperDataScience Podcast. I'm your host, Jon Krohn. We've got an outstanding episode for you today featuring the AI entrepreneur and many time author of bestselling AI books, Sinan Ozdemir. Sinan's latest book is his 10th book and it's called Building Agentic AI. Building Agentic AI is a hands-on book that covers everything you need to know to design, fine-tune, optimize, and deploy agentic systems effectively. And today's episode distills all of the most valuable tips and tricks from the book enjoy. |
| | 00:00:42 | This episode of SuperDataScience is made possible by Dell, Intel, Fabi, and Cisco. |
| | 00:00:51 | Sinan Ozdemir. Welcome to the SuperDataScience Podcast. You've been on here a few times. |
| | 00:00:56 | I believe this is number seven. |
| Sinan Ozdemir: | 00:00:59 | Jon Krohn, thank you for having me again. I would have to confirm that. And by confirm that, I mean ask Claude. |
| Jon Krohn: | 00:01:05 | I think you have the same ... It's perfect that you just did this because we're going to be talking about your new book, Building Agentic AI. And that book is the very first book from Jon Krohn's Pearson AI signature series. And it's available on bookshelves now. You've got a physical copy with you. I have a bunch of copies, but they're not with me. So I don't have them for recording to show in the video version of this podcast. But the reason why this is relevant is because everyone at Pearson and everyone at |

O'Reilly, where I know you do a lot of teaching, in fact, I know that you're in the middle of a three-week course that you're teaching at O'Reilly right now on ChatGPT. Everyone at Pearson and O'Reilly mispronounced my last name. And you just did it.

Sinan Ozdemir: 00:02:01 I just did it. I just did it. I wasn't going to say it because I was like, "I didn't know it was Chrome. Oh no.

Jon Krohn: 00:02:09 Yeah. So it rhymes with the bowel disease.

Sinan Ozdemir: 00:02:12 Was about to say like the disease.

Jon Krohn: 00:02:14 Yeah. And I said rhymes with, but it's actually just the exact same sound. So it rhymes with telephone. Anyway, I'm trying to stub this like this, but it's like there's an echo chamber of people at the publishers misusing it or mispronouncing it. And so I'll never win. It's a battle I can't win.

Sinan Ozdemir: 00:02:36 I'm not going to lie to you. It also doesn't help that it rhymes when you say it wrong.

Jon Krohn: 00:02:42 I know. I know. Maybe you should just give in. But we're not here to talk about me. We're here to talk about your book, which is outstanding. At the time of recording, you're in the top 10 in the artificial intelligence category on Amazon. Not the AI book new releases, not some tiny little subcategory, but top 10 in the AI category on Amazon. Full stop. Pretty cool. You're at like 4.7 star rating after several dozen reviews already. So we can quantify, not just because your book is part of my book series, but we can quantifiably say that people like this book. Congrats.

Sinan Ozdemir: 00:03:23 I appreciate it. No, thank you so much. In a lot of ways, I'm always proud of everything I write and I'm proud to be part of this AI series. And on the other hand, the way

sometimes I think about it, well, this is also my 10th book, writing about AI and agents for over 10 years now. And it's still refreshing when a new manuscript comes out and I see the reception and what's different, who likes it, the audience that's really resonating with it. So every time I release a new book, it's always just really fun to see who likes it, who's picking it up.

Jon Krohn: 00:03:59 Yeah, it's really cool. Yeah, I loved reading it when I was editing it and I'm so delighted that it's out now and we can be sharing it with the world. We could be doing an episode about it. And so what we're going to do in this episode is we are going to talk about some of the most important concepts covered in your book so that our listeners can, one, understand all these key concepts related to building agentic AI so they have them under their belt. And that will be enough. This episode will stand alone as a great episode, but also if they want 300 odd pages or more on this with lots of code, tons of hands-on examples, then they'll be able to reach for your book, Building Agentic AI.

Sinan Ozdemir: 00:04:45 Exactly. And soon there will also be a full Substack with all the case studies and a full video series with a lot of the case studies in it too. So whatever the medium you prefer.

Jon Krohn: 00:04:57 Exactly. I'm sure you'll have video course versions of it eventually in the O'Reilly platform as you often do. Yeah. Yeah. Perfect. All right. So let's get going. You're the start of the book in chapter one. You establish fundamental distinctions between different types of language models and you introduce the critical difference between workflows and agents proper. So yeah, you explain that like auto-regressive models are the writers. So things like GPT, Claude, Llama, while encoding models like Burt are readers. So these are the language models, but the key thing in building agentic AI isn't so much what kind of model people are choosing. It's defining the difference

between workflows and agents. And yeah, so tell us about workflows versus agents and what it means to have agency, what that distinction is.

Sinan Ozdemir: 00:05:48 No, no, happily. I think it's actually a very ... So almost a quick anecdote about that is the title of the book, Building Agentic AI, was actually not the original title. The original title of the book that I wanted, that I first came up with was along the lines of building applied AI. And I think that kind of evolved into agentic mostly because of that exact question, Jon. The question of, well, not everything I'm going to write about is a pure agent. And I always want to make sure I'm being honest and saying, some of this is agents, some of this isn't agents. So I wanted to make that distinction upfront as quickly as possible. The idea that not everything that is going to be built is a pure agent. So to your question, an agent, quite simply, is an LLM, a large language model with access to tools.

00:06:43 And I mean tools in the function calling sense, meaning this thing can Google something if it wants to. This thing can read a file from my desktop if it wants to while it's talking to me. That's almost a pure agent, if you will. A workflow is very much a deterministic data and code path. So most people's AI deployments on their production loads are workflows, meaning a question comes in and without asking an LLM, I use an LLM embedding to retrieve some documents. I construct a prompt that a human has already engineered. I run it through the LLM, I grab what the LLM says and I show it to the human. None of that was agentic. The LLM never decided to use a tool. It was all forced upon it. So when you take away the agency, I think that's kind of the key. When you take away the agency of the LLM to choose the next path, you are in workflow territory.

| | | |
|---|---|---|
| | 00:07:46 | Now there's also the hybrid case, which I'm sure we'll get into, but the distinction there is deterministic code paths versus agency among the LLMs. |
| Jon Krohn: | 00:07:56 | Right. So in the agency case, it implies that there's some amount of maybe a loop or the ability to call up other agents. Things are going to happen that you as the programmer that developed this system are not going to be able to predetermine. |
| Sinan Ozdemir: | 00:08:15 | Correct. And I think on some levels you can predetermine some things. For example, if I only give an LLM a tool to Google something, then okay, I have a pretty certain what it's going to do when I ask it to look something up. But a lot of modern deployments really start at five to 10 tools. And then there's the orchestration of tools, meaning which tool goes first? What if I use the tool wrong? What if I Google something inefficiently and then I have to follow it up with a second Google search? So you enter this kind of quagmire of it's not just which tools is it using, but in what order and how efficiently. And it's almost like you are inviting this nebulous workflow that the agent gets to decide how to navigate and you as the human kind of sit back and say, "Man, I hope this thing does it efficiently and correctly." |
| Jon Krohn: | 00:09:09 | Yeah. And so you, in addition to writing 10 books, that is despite that impressive total, not actually your day job really, you have built companies, you've built AI companies, you've been doing that for a long time, you do a lot of consulting. When you're advising somebody, say when you're consulting with a company, you're thinking about a product that you're building yourself, how do you decide where on the spectrum from deterministic to hybrid, you mentioned that word a few minutes ago, to fully agentic, proper a solution should be? |

Sinan Ozdemir: 00:09:42 Honestly, the first question I always ask is to a client, for example, is, "Describe to me the existing process." Full stop, and then I wait for them to answer. Half the time, they'll not respond. They'll stare at me and say, "There's some documents, that's a great..." I think Lindsay would probably know ... When they don't really quite know the process themselves, I go, "That's step zero, is you and I are now going to walk through what the existing process is for this task you are asking me to automate." That almost always, nine out of 10 times, will tell me if it's a workflow or an agent. The big differentiating factor is conditionals, meaning if we're walking through a step like ... I'll make this up. If you are trying to do a recommendation engine is always a fun one, meaning, well, when someone types in a product, I want to recommend something from my e-commerce store.

00:10:43 Cool. How does that work now? What do you do now? Do you base it on titles, descriptions, images? Walk me through that process now. If I go, "Well, it's really just a simple keyword search." Okay, great. Sounds like this is going to be a pretty simple workflow. There's no decisions to be made. There are no conditionals. If however, they say, "Well, if the user is under 18, we have to follow these set of rules because of the age restrictions in Ireland or whatever it is, and therefore these are the steps. If the user's over 18 but doesn't have an account, these are the steps." So when you start to get into all of these conditionals and rules, you enter more of the category of either a hybrid, which is, okay, maybe there's only three pathways. We just have to decide which pathway or if there's every step of the way, there is just one more conditional, one more gotcha, then it starts to enter the category of maybe an agent or some kind of multitude of agents is the right approach here.

| | | |
|---|---|---|
| | 00:11:42 | But almost always, an agent's work can be done through a workflow. It's just how big will that workflow be and is that workflow going to be easy to maintain? |
| Jon Krohn: | 00:11:54 | Great example there, crystal clear for me. I mentioned in my preamble, before I even asked you the first question today about your book, that you, in this book, you don't focus so much on any one particular LLM. You instead are equipping readers with everything that they need to know in order to be able to evaluate what the right LLM is for a particular circumstance in order to figure out how to configure their agents or their multi-agent system for their particular use case. So it's a book that pretty uniquely in the AI space has a long shelf life because it's kind of about these general principles as opposed to specific recommendations of what people should be doing. |
| Sinan Ozdemir: | 00:12:42 | It's very on purpose, very, very mindfully on purpose. I mean, you could argue that a lot of it's laziness in the sense that I don't want to update this book every year, so I can't be saying this is the definitive guide to GPT5, God forbid they make 5.7 or whatever it is that they're on, but it's also very much a machine learning engineering side of me saying, "To be clear, models will always change. The concept of an agent existed before ChatGPT. The concept of distillation and transfer learning existed before LLM." So this is just one more step in the process. So I very mindfully craft a lot of the case studies to say, the task is this. Whether the LLM changes tomorrow, you still need something, customer support, help to help people solve their hotel problems. So focusing on the task in the domain really helps ground it for a lot of people and again, keeps them comfortable in saying, "Well, don't worry, things will change. |
| | 00:13:45 | That's the point. This is how you work on this problem given X set of LLMs on your plate." |

| | | |
|---|---|---|
| Jon Krohn: | 00:13:53 | Yeah. And so despite all of this change that's happening, there is a bifurcation that you come back to throughout your book. And I even mentioned this in a question quickly earlier, and that's between auto-encoding models like BERT and auto-regressive models like GPT. And so that seems like something that is going to last. For decades, presumably, we're still going to have that bifurcation. It is something that it's worth people knowing. So could you explain this distinction between auto-encoding and auto-regressive models and when people should be considering using one or the other? |
| Sinan Ozdemir: | 00:14:26 | Yeah. I'll even preface that by saying technically that might not even be existing for decades. Even now with diffusion LOMs kind of breaks a little bit of the rules of what we consider a next token prediction. So auto-encoding versus auto-aggressive is very much the bifurcation of our time right now. Meaning when you talk to Gemini, DeepSeq, Claude, what have you, the chatbots of the world, they're speaking, they're generating text, they're generative AI. Now that next token prediction is autoregressive language modeling, meaning it is using the past, your prompt, to predict the future. That's kind of the hand wavy definition of that. The lower level is the LLMs are trained to predict the next token given context, given a prompt. Whereas an auto encoding LLM is trained to fill in a blank, not necessarily at the end using only past context, but anywhere in the sequence. |
| | 00:15:30 | If I give you a Wikipedia article and I take out a word in the middle and I say, Phil, what word was that? That's an auto-encoding language modeling task. If I take the word at the end off and say, "What word is that at the very end?" That is going to be more like an auto-regressive task. One of those is a generative model, the auto-aggressive, the other one is non-generative. BERT, the bidirectional encoded representation from Transformers, LLM, is an auto-encoding LLM. Can't |

speak, but it can do embeddings and classifications almost as well as a huge GPT model with only a hundred million parameters. So you get these size differences, but also these task performances are eerily similar even when they're thousands of times smaller.

Jon Krohn: 00:16:17 Nice. And that leads perfectly actually to my next question. You have no idea what my questions are, but the very next one is about parameter count. So you discuss in the book how parameter count gives a rough sense of the model's complexity and sometimes even its capabilities. And so you gave kind of an exception that proves the rule maybe in some ways there, where with a auto encoding model, you can get away with having far fewer parameters. And if you're not going to be doing a generative task, it can work really well for you. But let's say we're looking just at auto-aggressive models, at generative models within that class, your clauds, your GPTs, what have you, and kind of the LAMA family gives us a really good sense. We have so many different LAMA models to choose from. How can somebody go about guessing what kind of parameter count they need, what they should start with for a given task?

Sinan Ozdemir: 00:17:10 That's really the question of our time, Jon. It's really difficult because even if you stick in the same family, I actually like to think of Quen as a good family for this. Quen is a family of Chinese models from Alibaba, I believe. They have a much wider range even than Lama. They start at half a billion, like 600 million parameters, and they go all the way up to ... I'm going to Google this in real time because I'm curious.

Jon Krohn: 00:17:45 I can't confirm that it's Alibaba while you do that.

Sinan Ozdemir: 00:17:47 Okay, good. Yeah. I was like, I'm pretty sure. Over a trillion. So they go from half a billion to over a trillion. So that's a huge spectrum. GPT is over a trillion for

perspective's sake. So over a trillion is GPT level, claud level. Yeah,

Jon Krohn: 00:18:02 Like GPT5 level.

Sinan Ozdemir: 00:18:04 Yeah, like Opus level of size. Kimmy is probably another one up there in terms of open weight models that is on par parameter wise and performance wise. So when you think about the spectrum, you think, okay, bigger is better, right? Kind of. Bigger means more generalized usually, meaning when OpenAI makes a model, they need a model that's going to work for me in San Francisco and someone who speaks only Turkish and Istanbul and has different needs and wants throughout the day. So they need to think about that. So big doesn't always mean it's better at everything. It just means it can do more horizontal things even if there are things you don't need it to do. So when you're considering a parameter count, usually I think about it in terms of, again, quote unquote small, which is anything less than call it 10 billion parameters.

00:18:58 I call it small because reasonably on a machine with no GPU like my laptop with a CPU, I can reasonably run a compressed version of a model less than 10 billion parameters on a CPU. It's not going to be very performant, but it will technically work. Between 10 and 100 is kind of where you have your medium size and arguably you would break that down even more.That's where you're going to get more into your agents. An agent with less than 10 billion parameters, you can get away with things like looking things up, document retrieval, rag or web search, and you're not really going to get away with a lot of long horizon tasks, meaning tasks that require several tool calls, several turns of a conversation. In that medium range, 10 to 100, you can, especially if you're going to fine tune it with your task specific data, it's very likely going to be the case.

| | | |
|---|---|---|
| | 00:19:56 | When you need enterprise wide adoption, you need something that can handle multiple languages, multiple tasks, easy and hard. Usually you're in the large category, a hundred billion plus, both open and closer. And again, that's rough. |
| Jon Krohn: | 00:20:14 | That was great. That's the clearest kind of definition. I've never actually even heard anybody try to attempt so clearly how you can be trying to make some kind of rough finger in the air decision, a t-shirt size decision on what kind of model you should be going with. So that is very helpful. Thank you for that. |
| Sinan Ozdemir: | 00:20:29 | I hope so. And again, it's all an experiment. That's kind of the fun part in a lot of the book is chapter three, I think, is this literally called experimentation. The point of these is this is the fun part. This is the part where you get to try half a billion parameters versus eight billion parameters and prove that one is much better than the other. Putting your money where your mouth is, so to speak. And for me, that's a lot of the fun part. |
| Jon Krohn: | 00:20:52 | Yeah. Chapter three is a fun chapter and I have a number of questions for you from that coming up next. I just have one quick one to squeeze in first, which is related to context window size. So context windows are expanding rapidly, the number of tokens, the amount of context that a given model can handle in memory at any given point. And how does this change? We talked about workflows, deterministic patterns, agent design patterns. How does massively large context windows change those design patterns? |
| Sinan Ozdemir: | 00:21:25 | Yeah. So an agent, an LLM with tools, works in a loop, as you said earlier, meaning if you ask it a question, it's going to say, along the lines of, "First, I'm going to call this tool to do X, Y, Z tool call. Now I'm going to call tool two to do this tool call back and forth, back and forth." |

And then finally it says, "I have an answer for you, Sanon. It's X, Y, Z." As you keep talking to that agent, you add more tokens to the context window. As it decides to call tools, it's adding tokens to the context window. So when you're in the realm of agents, pure agents, LLM with tools, long horizon, meaning like just longer running tasks, require a longer context window. More importantly, they require a longer context window that the LLM can reasonably reason over. So there's another example called Needle in the Haystack in the book where I basically fill the context window of an LLM with random facts, trivia facts.

00:22:27 And then at certain points in the prompt, I inject my own birthday, not a very public piece of information that I expect an LLM to know. So I'm reasonably confident it wouldn't know my birthday unless I told it my birthday. And depending on where I put it in the context window and how much the context window was filled, it would not see my own birthday, even though I clearly put it at some point in the context window. So it's not even just about can the agent get a long context window. You also have to consider, can that LLM, is it powerful enough to actually see everything in its own context window consistently? And it's almost a paradox, right? To say, "I put it in the context window. Why can't it see it? " It's just a fact of nature with some LLMs.

Jon Krohn: 00:23:18 Yeah. I mean, it's astounding to me that it can do this at all. I mean, basically all of this technology completely blows my mind. I did a PhD in AI that I finished in 2012, and if you had asked me if in my lifetime we would have machines that could search over millions of tokens and have these lucid conversations with us, I would have said, "I don't know if we will have that in our lifetimes. That seems really, really hard." And that it works most of the time is pretty mind blowing.

Sinan Ozdemir: 00:23:46 To be fair, in five years, they were about to invent the architecture that would enable it. So at the time, I think it was a fair answer.

Jon Krohn: 00:23:56 Yeah. Yeah. I'm still constantly blown away by the kinds of things. Now, things like NanoBanana and the kinds of things that we can do with that are so crazy. It is a quickly rapidly evolving and surprising world, but it makes it a fun- It does. ... fun area to be part of. Just really quickly on the needle in a haystack test, part of what makes it so hard, I think, to evaluate the needle in a haystack, because you gave the example there of your birthday. Imagine if we took your textbook, and so the whole textbook is about building agentic AI, and then we insert your birthday kind of randomly at one point in there. That might be very salient to an algorithm. It might capture its attention because of how unusual it is in there. And so these needle in a haystack tests, I know that recently people are trying to come up with ways of making it kind of more in- Yeah, harder than that, because it's such a tricky problem, but all the frontier labs are tackling it, and we're getting more and more impressive results all the time.

Sinan Ozdemir: 00:25:00 Oh, like two years ago, it was atrocious. Anthropic is very good about this. They actually published their own needle in a haystack example. They were terrible at it. Now they're great. That's why they posted it because they said, "Look how much better we got." I think in my book, I unfortunately had to call out Grok with a K because it was the worst that needle in the haystack, but I was like, "This is what it looks like when it can't see its own context window halfway through."

Jon Krohn: 00:25:25 Yeah. And so speaking of experimentation and research, let's jump to that chapter three that you mentioned earlier, the fun one. I mean, there's lots of fun chapters, but chapter three is a particularly good one. And in it,

you present your comprehensive framework for AI evaluation. So you emphasize that accuracy alone isn't enough. And so you introduce multiple metrics across different task types, retrieval, classification, generation, and you stress the importance of reproducible experiments. You conclude the chapter by noting from now on, we will be incorporating evaluation language into every case study. So then throughout the rest of the book, you have these fantastic detailed case studies that build on each other and you use this common evaluation language that you introduce in chapter three throughout, which is brilliant. So when you organize this evaluation by task buckets, generation, multiple choice, embedding, classification, why is that separation so critical?

Sinan Ozdemir: 00:26:26 Yeah. So the split is usually of the types of LLM tasks, there's generative and understanding. And then under generative, there's a multiple choice and free text, meaning it's basically like auto encoding versus auto-aggressive is how I try to think about that analogously. Meaning if you're talking to a chatbot or an agent, which is just a chatbot with tools, you're asking it either to produce a paragraph, a sentence, several paragraphs, whatever, free text, or you're asking it to pick from a set of options. Should I proceed? Yes or no? Is this good enough to post on LinkedIn? Yes or no? That's multiple choice. I'm basically collapsing the entirety of this deep learning architecture into a binary classification task versus understanding tasks, which are embeddings and classifications, which are similar to multiple choice, but just with a different architecture. Each one of those has their own suite of metrics because how I evaluate a child's essay on a cashier on the eye is going to be different than how I evaluate the embeddings that this embedding model is producing.

00:27:39 They're just not the same task. They're not built for the same thing. They're all at LLMs. OpenAI embeddings are

produced by LLMs. Classification models are run, for the most part, by LLMs. So the evaluation is less on the model. It's more on the task that you're trying to perform. And whether you're performing classification through any kind of architecture, my book from 10 years ago, Principles of Data Science, talked about accuracy, precision, recall, sensitivity versus specificity. I also talk about that in my book from two months ago, Agenttic AI. It's the same classification that I'm asking an agent to do. It's the same task. It's just a different model is now doing it. So evaluation is tricky. It's the longest video I ever wrote or made was like nine, 10 hours on the O'Reilly platform was evaluations because there is no one size fits all. It's what are you doing?

00:28:34 I'm now going to walk through 20 case studies that are all very different from each other, all with different metrics.

Jon Krohn: 00:28:40 Yeah. And so to dig into this a little bit more, you mentioned there are all these different kinds of metrics for evaluating performance. So I already said in a question a few minutes ago how accuracy isn't enough. In your book, you emphasize how using precision recall and for something where you're trying to rank results, something like mean reciprocal rank, MRR, using those metrics together because each exposes different failure modes of a model. Do you want to tell us a bit more about that?

Sinan Ozdemir: 00:29:10 Yeah. So precision recall is probably the more, I would say, usable metrics across for most people, meaning ... I'll say it this way. If you ask an LLM, you give it a LinkedIn post and you say, "Is this going to get a lot of engagement on LinkedIn?" And it says, "Yes." Okay, great. You post it. It doesn't get a lot of engagement. That model had a false positive. It told you yes, but really it was no. When you care about false positives a lot, when they are expensive to you, you care about precision. Precision is the

measurement of all the times the model said yes, how often can you trust it? So when the model says yes, Go ahead. How often is it correct in saying yes? That's precision. So when you care about false positives, precision is your metric. Recall is kind of the opposite.

00:30:11 Recall is of all the times it should have said yes. How many times did it? So if false negatives are expensive to you, recall is the metric you care about. Because if the thing says this is a terrible LinkedIn post, but you post it anyways and it gets a lot of engagement, that's a false negative. It didn't want you to post that. And recall is a measurement among other things. A recall is effectively a measurement of how many false negatives that you're seeing out of this system. And that was a pretty dense explanation for two, honestly, one of the simplest metrics in machine learning. And it kind of goes to show that the conversation around evaluation is not always as simple as, here's the fraction that you care about. It's no, before we get to math, what do you, the human care about? What's expensive to you?

00:31:00 If you say this factory part off the line is good, but it's not good, is a plane going down or is someone's light going to break? How expensive is a false positive to you? If it's expensive, precision is the thing you need to look at. Recall shouldn't matter as much. I'll happily throw a part away on accident. At least if I know everything off the line is going to be right, precision matters the most. So again, it always comes down to not just the task, but even the risks of failing that task.

Jon Krohn: 00:31:33 That was a really impressive explanation of precision and recall. And you must have taught this a lot of times or you're just a lot smarter than me because anytime I'm talking about precision and recall, I always bring up a chart. I'll quickly do a Google image search just to make sure that I have that I'm getting things right because

there's something about it that it's not intuitive. I hear you. Accuracy. The word accuracy and what that means in machine learning, you can kind of intuitively kind of guess it. It's pretty straightforward. But precision and recall, like those are words that we use in English, but they don't translate very accurately and very precisely to the meaning in machine learning. They don't. Yeah. It's something that I'm just always like ... Yeah.

Sinan Ozdemir: 00:32:18 They don't. And I didn't even mention that. I'm pretty sure sensitivity and recall are just synonyms. It's literally the same metric. I've done this a lot, Jon. In 10, 15 years ago, I was also looking up that. And for some of you listening, you're going to picture this immediately in your head. That square with the circle inside of it with the line down the center where on the left is like true positives on the right is the false negatives and there's a colored in, red and green. I still see that image in my head of like me Googling it 10 years ago, and that's how I remember it.

Jon Krohn: 00:32:52 Cool. So once your readers or our listeners are equipped with the evaluation metrics, kind of evaluation frameworks, in chapter four and five in the book, you start diving deep into agentic systems themselves. And you're notably balanced about agentic systems limitations. You emphasize that some of the most powerful AI applications are combinations of predefined workflows and agentic behaviors. So kind of going back to that spectrum that we were talking about earlier on in the episode. And you caution that without a predefined pathway, you would need a sophisticated auditing system to make sure that everything's on track. Do you want to tell us more about that?

Sinan Ozdemir: 00:33:35 Yeah. So as I was kind of hinting at earlier, the two types of AI applications, workflows and agents, obviously there's almost always going to be some kind of a hybrid. There's the agentic workflows, if you will. The easiest

example I can always think of is deep research. Whether go to OpenAI right now, not you, not right right now, but go to OpenAI, turn on deep research if you have access and ask it anything and be very, very clear. Do not ask me any follow-up questions. This is all the context I'm going to give you. Do not ask me any questions. Please start the research now and tell me when you're done. It will always 100% of the time ask you a follow-up question. That's an identic workflow if ever I've seen one. They are forcing a node in a graph. They're not telling the agent to do anything.

00:34:27 The agent could have been prompt injected to bypass that step, but OpenAI is forcing a speed bump to ask a follow-up question before it gets to the actual agentics part, which is the research. So when you combine those two things, and usually the kind of speed bump is the most common way to do it, saying, "Well, before we get to the agent, let's clarify. Let's make sure this is the right thing." It's almost like if you use cursor, anyone, the coding environment with AI built in, it's almost like the planning mode is when you turn on planning mode and then turn it back to agent mode, you are basically doing the agentic workflow yourself. You are deterministically making it plan and then you tell it to go implement it as an agent. And when you do that, you're really filling in the cracks of what can be considered the edge cases, meaning, well, you should have asked follow-up, again, false positive versus false negatives.

00:35:27 OpenAI is happy to say, "Whatever, man, I'm going to ask you a question and you're going to deal with it. It's fine because it's more expensive to us, meaning people will complain more when it doesn't ask a follow-up question and we waste their time and money doing research they didn't want. " So even for them, it comes down to false positives versus false negatives. And solving that with a

workflow can be one way to basically mitigate those issues with an agent

Jon Krohn: 00:35:53 Deployment. I love that example. And I also am a big lover of OpenAI deep research.

00:35:58 Oh, great. Yeah. And I actually particularly, I use all of the major Frontier Labs chat interfaces. So I use Gemini Pro, I use the paid version of Claude, and I use the ChatGPT research. And you start to get this sense of which application's going to be best for some particular use case as you go on. And it is surprising how unequivalent they are, that there's some tasks related to reviewing a podcast episode or preparing for a podcast episode where I'm like, "I will not do that in Gemini. This is definitely a Claude task." And every once in a while you try it out because you're like, "Oh, the Gemini is top of the leaderboards right now. I should be trying that out. " And you're like, "No, still Claude."

Sinan Ozdemir: 00:36:42 Question for you, why? What about the output of Gemini do you not like in that instance?

Jon Krohn: 00:36:50 Yeah, yeah. So somehow it's the ability of the algorithm to guess what kind of structure I'm looking for in my output, despite me not having provided enough relevant context for it to guess that. And I think Claude is at the time of recording this episode, the leader at that. Somehow I can give, with large amounts of context, I can get Gemini or ChatGPT or Claude to do what I want, but Claude more often than not, I just type a couple of quick sentences with very minimal context and it spits out an output and I'm like, "That was exactly what I was looking for. Thank you.

Sinan Ozdemir: 00:37:38 Okay. Interesting. Yeah. I'm curious to how much of that is like it was coincidentally the way Anthropic architected

it is what you like versus does the LLM's preference happen to align with yours? I'm just curious.

Jon Krohn: 00:37:55 Yeah. So maybe somehow the kinds of tasks that I end up doing, and actually a lot of software developers prefer Cloud CloudCode. And a lot of the tasks that I'm doing are kind of adjacent to machine learning or AI or programming in some way. And so maybe there's just a lot more relevant training data for the kinds of problems that I'm dealing with.

Sinan Ozdemir: 00:38:20 Yeah. And it does come back to a lot of when you're picking a model. I mean, experimentation is a huge part of it, but even just having that kind of gut intuition of, well, Anthropic spends more money on throwing more code examples at Claude. I mean, GPT does kind of, but they have codex. They have a separate LLM codex to do coding than they usually have GPT5 do. So when you talk about which model do you use, no one usually says GPT5 codex, they say Anthropic because it's both the chatbot, the thing that can talk to me and the thing that can write code.

Jon Krohn: 00:38:54 And right now, another definite thing is if I'm going to be doing some kind of image manipulation, I'm going to nanomana. It is amazing the failure modes though still. Obviously we're very early in this technology. And as we said early in this episode, it's astounding what these models can do at all. But it is amazing sometimes the ways in which they fail. And that's actually, that's a key thing with agent design and evaluation, isn't it? Coming back to your building agentic AI book, the ways in which LLMs, even the most cutting edge LLMs, the biggest amount of parameters- Smartest. Yeah. The longest thinking times before they output something, the kinds of mistakes aren't the same kinds of mistakes as a really smart human would

Sinan Ozdemir: 00:39:45 Make. Agreed. I think, I mean, yes, I wouldn't even give an example. A simplest example I could think of, another case study is, I forget what chapter it is, but basically I give GPT 4.1, which was the latest model at the time of writing, I give it a tool to look up Airbnb policies and say, "You're an Airbnb policy bot. You answer questions given Airbnb policies." I will tell literally one sentence in the system prompt, always find a relevant article before answering the question. 5% of the time in that case study, GPT 4.1, not nano, not many, just normal GBT 4.1. 5% of the time when told to always look up and find articles, never even once called the tool and just gave an answer for no reason and made it up. I was like, I'm not only talking to arguably the smartest LLM on the planet, I gave you one instruction.

00:40:46 Always look it up before answering the question, and yet you still can't do that 100% of the time. Would a human do better or worse? Depends on the human, but a decently well-paid human will do it 100% of the time and not get fired.

Jon Krohn: 00:41:02 Yeah. That is a great example. I love that. And that's one of the examples that I saw you provide in a live workshop at the Lightning AI office in San Francisco. So I have this fellowship of lighting AI, and so I do stuff for them. And you've been roped into some stuff for them as well.

Sinan Ozdemir: 00:41:20 I was about to say, you got me into it. Yeah.

Jon Krohn: 00:41:23 Yeah. I roped. And you did this astounding presentation. It was so entertaining. It was like infotainment. I got to say, that talk that you gave that night in San Francisco, if any of you listeners ever have the opportunity to experience Sinan Ozdemir speak online, even tons of things in the Rally platform, but speak in person, there's this energy. When you're standing up and you're delivering jokes one after another, it's like, I'm laughing,

I'm learning. It was one of the best experiences of my life, period.

Sinan Ozdemir: 00:42:00 Damn, man. Well, thank you. Sorry. I didn't realize you were going to say that. That's really nice of you. I mean, I care a lot, man. It's supposed to be fun. Learning is supposed to be fun. This is not what we're always told. And people not having fun in school and people hating their math teachers. I was like, "That's BS, man. Math teachers can be fun. AI teachers can be fun." Just because I have a master's in theater called mathematics doesn't mean I can't crack a joke on stage while I'm talking about positional bias in LLMs. So I try really hard to keep it engaging, keep it relevant, keep it grounded, just keep it light as much as I can.

Jon Krohn: 00:42:39 Yeah. You do a great job of it. And it's wild because you're actually, you're a young person and you've writtenenes. You do all these lectures. Yeah. I mean, it's astound you to think where your career will go. Anyway, back to your book, back to building Agentic AI, let's fast forward a little bit to kind of another exciting moment in your book, which is chapter seven, where you present some counterintuitive findings, the kinds of counterintuitive findings you would present with a lot of excitement at an in- person lecture. And so you do benchmarking with reasoning models. So using the Math QA dataset, humanity's last exam, and you found a no obvious correlation between the level of reasoning and the LLM's performance. That is surprising. What do you make of that? Tell us a bit more detail about this experiment and the surprising results.

Sinan Ozdemir: 00:43:34 Yeah. And it's actually a pretty consistent result. I know that if you look up benchmarks, that's the first thing someone's going to say when they hear this is, "Well, no. If you look at the benchmarks from Anthropic, clearly the reasoning is going to lead to better results." Yeah, there

are a lot more benchmarks out there that Anthropic has time to inform you of. And if you run them on the other benchmarks they're not reporting, you are going to see the opposite. In fact, in a later chapter, I also do other benchmarks with reasoning. And even if they do show a correlation, it's maybe one or 2% increase in accuracy and a 2X increase in cost. So on one side of the coin, the AI does not get better at the task when you add more reasoning. On the other side of the coin, even when it does, it almost never, in my consulting, outweighs the cost of having that reasoning turned on and you are better spent that time prompting, few shot engineering, building your own kind of reasoning traces that are smaller, shorter, and more succinct in what you are trying to do.

00:44:43 So it's strange. And again, like I said, it's not always the case that reasoning doesn't yield better results, but it's really important to know that it is not a 100% correlation that when you turn on an increased reasoning, performance at a task will get better. That is not a guarantee.That's all I want to say. It's not that it's not a guarantee, or rather it's not that it's not going to happen, is that it's not a guarantee.

Jon Krohn: 00:45:09 Yeah. And so the key thing here that we're, in case I didn't make this explicit when I brought up this topic of reasoning models, what I mean is, so the first kind of really well-known reasoning model was o1 from OpenAI.

Sinan Ozdemir: 00:45:22 I would argue it's R1 from DeepSeek was the first reasoning model.

Jon Krohn: 00:45:25 I guess, but that people would probably have used ... I mean, maybe amongst our listeners, R1 might be something that people use all the time, but in the general public.

Sinan Ozdemir: 00:45:35 But you're right, you're right.

Jon Krohn: 00:45:36 And yeah, so those reasoning models, they spend time before just spitting out an output. So where in your typical ChatGPT setup or cloud setup without reasoning, you have this stream of consciousness spit out immediately. The reasoning models were the first time that you weren't getting that where behind the scenes this processing is happening where it is still just generating output, but it's reviewing that output. It's using it to come up with summary points and to consider different options. And so you can wait huge amounts of time depending on how you set up the reasoning model to work, you can wait huge amounts of time before anything is output to you as a user on the screen. And so in your book, you mentioned that your reasoning benchmark evaluation took five hours for just 1180 LLM calls and that- A

Sinan Ozdemir: 00:46:34 Lot of money, to be clear, it's very expensive.

Jon Krohn: 00:46:37 It can cost a lot of money for sure. And in contrast, 180 LLM calls, if we were doing immediate output results, not using a reasoning model, you could have done 1180 LLM calls in seconds or minutes.

Sinan Ozdemir: 00:46:52 Oh, absolutely. Yes. And to be clear, the reasoning outputs, to your point, are just ... It's the same stream of consciousness. It's just you don't get to see it. OpenAI or Anthropic and Gemini, when they make these reasoning models, they train the models to perform this reasoning, but it's all just auto-aggressive next token prediction. It's just that they've done it in a format in which the UI can basically hide it from you until the model basically raises its hand and says, "Okay, I'm ready now. I can actually talk to the user. I'm done thinking." So you can induce a level of reasoning like that simply by saying, "Think through the problem before answering the question." The

way that it gets better over time is how you train the AI to produce those reasoning steps, meaning what methods are being used to teach the LOM to produce that reasoning.

00:47:48 One of the funnier things you can do with a reasoning model is literally just say hi to it. And I almost put this case study in the book where for every reasoning model, at every reasoning effort, like how much reasoning, all I wanted to do was just say hello and see how much reasoning it produced just to say hi back to me because sometimes it would produce a whole paragraph. There was one model who I won't say out loud who produced three paragraphs of reasoning very genuinely along the lines of the user is saying a question, but I've been trained to do this. Maybe they're trying to ask me a task under the question, maybe they're trying to do this. For now, I'll just say hello and see what they want. And then it says, "Hello, how can I help you today?" I was like, "Oh, I waited 30 seconds for you to say hello to me."

Jon Krohn: 00:48:37 That's funny. You have a lot of creative ideas for tests which get born out in the case studies that you have throughout the book. Let's jump now to the end of the book. So you have nine chapters in the book, chapters eight and nine, both focused on taking AI to production through things like fine tuning and optimization. You emphasize on this note just now of being able to get performance for at high speed, at low cost, that's what we're ultimately looking for most of the time as someone who's engineering an AI system. So you emphasize that fine tuning enables cheaper, faster, and more trustworthy models through baking in domain knowledge and aligning confidence with accuracy. You cover things like Laura, low rank adaptation, quantization, distillation, and domain adaptation. And then in chapter eight, you note that optimization is about trade-offs. So these kinds of things, balancing cost, speed, accuracy, and privacy.

00:49:29 When our listeners or readers of your book are quantizing models, reducing them from 32 bits to four bits, or distilling them to get them from a hundred billion parameters down to 10 billion or whatever. What are the kinds of performance hit that practitioners should expect and what are the breaking points?

Sinan Ozdemir: 00:49:50 Yeah. And whether it's quantization, which is the reduction of precision. So an eight billion parameter remains an eight billion parameter model just with smaller amounts of memory required to hold the numbers or distillation, which literally will try to transfer knowledge from a big model to a small model, or even using low rank adaptation, Laura, to basically hijack the fine tuning process and fine tune fewer parameters. You are almost always running the risk of saying, "Well, I'm doing this for a reason. I need less memory. I have less money than OpenAI. I need to do this faster, cheaper, but definitely not better." So in one of my, it's actually not in the book, but in my newsletter, an example that I do, it's in another book that I write, is I do a quantization of Llama and I not only benchmark it just to say, how is it differing on benchmarks?

00:50:49 And it did a little bit worse on benchmarks. I would even quantify how many tokens of the ones it's considering are the same. Meaning if I look at the top 10 tokens at every step of the way, how often do they not have the same tokens in common with each other? So it comes to a point where it's almost a different LLM. It's almost to a point where when you quantize a model or when you distill a model, you're effectively talking to the cousin of the model that you were talking to. It's not exactly the same. It's not too dissimilar, but there's definitely differences you're going to notice. And it's almost always going to be that it got a little bit less intelligent on benchmarks, and more importantly, it's just not going to produce the same

outputs that you were expecting. And a lot of people get that kind of misconstrued.

00:51:42 They're like, "Well, quantization's like magic. I can finally, now this thing fits on my CPU." But when you look at it side by side, the non-quantized versus the quantized version, then it becomes really obvious what the differences are. But when you're only looking at the quantized version, it's still legible, it's still human readable, it's just you're not really getting a chance to see it at scale.

Jon Krohn: 00:52:04 Awesome answer. And like so many things in this episode, you've brought a lot of clarity to complex topics that could be nebulous for a lot of listeners or maybe even foreign to many of our listeners. A final kind of question related to your book that I have for you is while you were writing it, was there anything that surprised you, like some experimental results or something that you learned that was maybe wildly different from what you were expecting?

Sinan Ozdemir: 00:52:32 Oh, I wasn't expecting the one you mentioned before, the reasoning, the non-correlating reasoning to performance, I had a hunch. That was very much a hunch from a client that I had been working with who was insistent on using reasoning. And I was showing them basically doing an experiment of low versus high and I was like, "Something must be wrong." I'm getting basically the same answers on low versus high. And so I tried it again and then I realized, let me do this in my book. Let me get other benchmarks, other LOMs, how consistently can I reproduce this? So that genuinely was actually pretty surprising to me that it was that inconsistent or rather that consistently non-correlative between reasoning and benchmark performance. But to give you another example, just for the sake of talking about more case

studies is in another ... I have the book on my other screen here.

00:53:36 In another example where I fine tune a Quen model, I also do what's called speculative decoding, where the idea is I first ask a small model to do something and every couple of tokens, it stops and it asks a big model to basically double check the work. So you're basically hoping that the small model is smart enough more often than not, that the big model doesn't have to really do much except saying, "No, you're good. Keep going, keep going, keep going. " And then when it says, no, that's wrong, the big model takes over for a couple tokens and it goes back and forth. I was actually quite surprised to see how consistently I could correlate what types of questions that I was asking versus how much speed and memory performance I could get. Meaning, said another way, I was starting to be able to actually diagnose and predict, "Oh, I bet this question won't benefit from speculative decoding, even though speculative decoding is supposed to be this really magical thing that makes LLMs more efficient." But I was starting to be able to guess which ones were not going to work.

00:54:50 And that really kind of got me thinking that, okay, well, this is kind of how routing works in a lot of ways. Meaning, how does OpenAI know whether to talk to the slow version of the model or the fast version of the model in real time? And I'm sure our viewers can attest, it doesn't always get it right. Sometimes it should have gone to the slow one, sometimes it should have gone to the fast one. But when you start to make those connections between, "Oh, this type of task won't benefit from this versus this type of task will benefit from this, " you start to really be able to get more confident in your understanding of that material. So I was actually quite surprised by the end of it when saying, "Wow, I didn't realize it would be so clear that these types of tasks

benefit and these other types of tasks don't." And it's a really great feeling when it happens, I'll admit.

Jon Krohn: 00:55:39 Awesome. I love those surprises that you discovered as you were working through your book. And yeah, I'm sure there were a lot of interesting tidbits, surprising tidbits for our listeners today in this episode. Maybe you can hold up your copy of Building Agentic AI again. So don't hesitate, go get Building Agentic AI from Sunan Ostermer right now. It is an outstanding book. And yeah, thank you, Sanan, for taking the time out of your absolutely packed weeks. You do so much teaching, consulting, and writing. It's a joy that you have made time to be with us on this podcast so many times as well.

Sinan Ozdemir: 00:56:20 Jon, Dr. Crone, if you will. Jon, I'm always happy to be here. You know that. I love writing, and as many of your viewers at this point know, for me, writing has become my proxy to teaching. I used to be a full-time teacher. So for me, making these books is just my way of asynchronously teaching. Everything I learn from my clients, I learn from being on stage and talking to people at conferences, they go into here. Obviously, everything's anonymized. No one is ever going to know, but the things that I learn through helping people with LOMs over the past decade, they end up in these books. And for me, that it's such a joy to hear people say, "Oh, I know not exactly this, but kind of like this was happening at my company and I was able to translate it. " That's the point.

00:57:08 It's not going to be one-to-one exactly what you need, but it's hopefully it's enough to spark the … That's similar enough to what I'm doing at my job and in my line of work that I know how to adapt it. And that's when I get really happy.

Jon Krohn: 00:57:22 Fantastic. Thank you, Sinan. And I love that perspective. I hope too to be able to … I've just started writing my

second book recently, and I hope to someday be able to have the prodigious output that you do as well. It's really remarkable because I also, I benefit so much and everyone that I interact with, not just readers, but like you say, clients, probably podcast guests or podcast listeners benefit from me learning through the process of consolidating information into a book. And yeah, so you're an inspiration for me.

Sinan Ozdemir: 00:57:57 Well, I appreciate it, man. It's quality over quantity sometimes. So I'm excited to read your second book too. The first one was amazing. Still is amazing, quite frankly, and I am excited for the second one.

Jon Krohn: 00:58:09 Nice. Yeah, yeah. We'll see. Maybe 2026, it'll be out, hopefully. And yeah, so for people who want to have more of your thoughts before the next time that you're on this show, which probably won't be in the far too distant future, how else should people be following you? I know you have a great podcast of your own. I will appear on that podcast. As soon as my book's out, then I'll be doing a podcast tour as well, including yours. Your top of the list.

Sinan Ozdemir: 00:58:35 You heard it here first, folks. Yeah, I have a podcast practically intelligent. I have that with my co-host who's a VC guy who's my former student. I also have a Substack with the case studies from my book coming out, about one case study a week. You can find that on my website, SinanOzdemir.ai, which all of this you can find on my LinkedIn. They all cross reference each other. There's not a lot of Sinan Ozdemir out there, and there's very few of them who are in AI. So you're not going to have trouble finding me. Don't worry.

Jon Krohn: 00:59:06 Yeah, yeah. You own that SEO by now for sure. Fantastic. Thank you so much for taking the time and we'll catch you again soon.

Sinan Ozdemir: 00:59:14 Thank you. Bye.

Jon Krohn: 00:59:17 What a terrific and fun episode as always with Sinan Ozdemir, in it Sinan covered how an agent is an LLM with access to tools that can decide which tools to use and in what order while a workflow is a deterministic code path where the LLM never chooses its next action. To decide if you need an agent or workflow, walk through the existing process and count the conditionals. If there are many branching decision points, you're likely in agent territory. In terms of LLM parameter counts, he had great guidance for us on three different tiers, the small tier under 10 billion parameters, which can run on a CPU and handle simple retrieval tasks, medium size, which is 10 to a hundred billion model parameters in enabling multi-turn agentic tasks and large, that's 100 billion parameter plus LLMs, which are needed for enterprise wide, multilingual deployments and things of that complexity.

01:00:11 We also talked about how accuracy alone isn't enough for evaluation. Precision matters when false positives are expensive and recall matters when false negatives are expensive, and we got insight into how scaling reasoning time upward doesn't guarantee better performance. As always, you can get all the show notes, including the transcript for this episode, the video recording, any materials mentioned on the show, the URLs for Sinan Ozdemir's social media profiles, as well as my own at superdatascience.com/959. Thanks, of course, to everyone on the SuperDataScience podcast team, our podcast manager Sonja Brajovic, media editor, Mario Pombo, our partnerships manager Natalie Ziajski, our researcher, Serg Masís writer, Dr. Zara Karschay, and our founder Kirill Eremenko. Thanks to all of them for producing another fantastic episode for us today for enabling that super team to create this free podcast for you. We're obviously grateful to our sponsors. You can

support the show by checking out our sponsor's links in the show notes.

01:01:05 And if you ever use one of those sponsor products, let them know that you heard about it on this show. That would go a long way to helping us out. And if you yourself are ever interested in sponsoring an episode, you can find out how to do that at JonKrohn.com/podcast. Otherwise, share, review, subscribe. But most importantly, just keep fun tuning in. I'm so grateful to have you listening, and I hope I can continue to make episodes you'd love for years and years to come. Till next time, keep on rocking it out there, and I'm looking forward to enjoying another round of the SuperDataScience Podcast with you very soon.