AI Evals For Engineers: Course Preview (Chapters 1-3 of 8)

63 MIN · YOUTUBE · HTTPS://WWW.YOUTUBE.COM/WATCH?V=JJMYWFQYLY0
https://www.youtube.com/watch?v=jJMYWfQYly0

SUMMARY

The discussion revolves around a live workshopping session focused on the evaluation (eval) of large language models (LLMs), featuring insights from Eugene, a senior applied scientist at Amazon. The conversation covers the challenges of evaluating LLMs, the importance of error analysis, and the iterative process of refining prompts and evaluation metrics.

- *The session emphasizes the difficulty of conducting effective evaluations for LLMs, particularly in understanding their limitations and capabilities.*
- *Eugene identifies three critical "gulfs" in the evaluation process: comprehension (understanding data), specification (writing effective prompts), and generalization (ensuring models perform well across various scenarios).*
- *The importance of comprehensive evaluation metrics is highlighted, especially for applications with wide customer impact.*
- *Error analysis is discussed as a vital step in the evaluation life cycle, involving the identification and categorization of failure modes.*
- *The conversation touches on the iterative nature of the evaluation process, where insights from error analysis inform improvements and measurement strategies.*
- *The use of LLMs for synthesizing failure modes and clustering annotations is suggested as a way to enhance the evaluation process.*
- *The need for a balance between human involvement and automation in the evaluation process is emphasized, with a focus on making analysis more accessible and efficient.*
- *The session concludes with a discussion about the future of evaluation tools and the potential for new methodologies to streamline the analysis and measurement of LLM performance.*

Hey guys. How's everyone doing? Great. I have a question. How can I check it on the YouTube just to make sure it's actually good. yeah. Let me give you the link. I'll DM Can I Can you share a link to book, please? Wow, people are already asking for links to the book. Take our course. Take our course. Yep. We're still writing it. That was a workshopping session. So this is our first time doing a live workshopping session of this. So I am slightly nervous that Eugene is going to be like, "This is horrible. Never ever Whatever.

but yeah, I'll maybe go over high-level what we're going to do. so this is We know that you all don't have the book. but what we did was we told Eugene, "Please read the first three chapters. Let us know what sticks, what doesn't stick." And we have a bunch of targeted questions. You know, Eugene is a very senior possibly even higher title than very senior applied scientist. Oh, did we lose Hamel? Oh, no. Now he's back. You're back. All

right.

No, you're good. I was introing Eugene. So Eugene is a good friend of ours, very senior applied scientist at Amazon. and what I admire most about Eugene is how willing and open he is to share his thoughts on things that, you know, he himself is also learning with the community. He has a very extensive blog. I have learned a lot from it. Everything from, you know, I started following Eugene back when he was writing about recommender systems to now when he's writing about LLM as a judge, to evaluations, to you can't measure what you can't improve what you can't improve what you can't measure, you know, the whole There's It's a great journey to follow. So I highly recommend checking out his blog.

so today, I know I have a bunch of questions prepared for Eugene around kind of his experience with some of the failure modes and some of the frameworks that were presented in the first three chapters. maybe was I'll start with a very brief high-level overview of kind of what's three chapters are about. first we kind of talk about what is evals and why evals are so hard at least in our experience. why is it different from traditional ML or MLOps, traditional software software engineering. Then we go into a little bit of thinking about how to work with an LLM. So how does an LLM work as a black box? How do we reason about its strengths and weaknesses? How do we use it effectively? and then kind of some getting our feet wet into thinking about what does an eval mean when we're building applications? what kind of metrics would we want? and how might we get some of the building blocks to compute those metrics, for example, labels?

and the third chapter that Eugene also skimmed over is the very first stage of the evaluation life cycle, which maybe we'll go through in the diagram. first you do an error analysis, analyze, then you measure things at scale, then you improve based on your measurements. so Eugene and we will discuss kind of the process of error analysis from looking at your data to organizing that into structured failure modes and iterating on those over time.

So without further ado, I'm going to move to my questions. And I'm going to hit Eugene with the first one. Eugene, you ready? Yeah, go for it. Okay. So thanks for taking your time to read and review, and you're the first one doing this. So I'm curious from your perspective at Amazon, what are some of the most pressing challenges in LLM evals that you see on a day-to-day basis? And do you feel like these chapters touch on these well so far? And what parts of them might you resonate with?

Yeah. Well, thank you, Shreya. It's a huge honor to be one of the first few to read the book. it really resonates with me. I think the main challenge with evals that people just don't do them. they don't know how to do them. And when you look at a lot of the biggest issues that come up that gets you negative press or has customer impact, some of the biggest issues is just not having evals to know how to even write the prompt well.

Mm. Some of the biggest issues is just not having evals to know that the model cannot actually do the thing. Well, the current generation of model can't do the thing, and maybe what you're trying to reach is way too far ahead. You actually are blind on that. And then the other thing is that maybe you have evals on a very small surface area, like the common stuff. But when you get to the tail, especially on something that has very large

scale, you actually don't have insight. This doesn't mean that you have to have very comprehensive evals, but you probably needed to be more comprehensive than you think right now, especially you have wide customer impact.

Totally. I've also resonated a lot with long tail and inability to specify prompts well in the first place. so why don't we go right into chapter one? My first question about that kind of being, you know, this Okay, I'm going to scroll through to kind of where I say where you can use evals for. I actually do my scrolling up to the gulf three gulfs. Yeah, tell me about that first. This really resonated with me. I actually didn't think of it this way.

And I I think that understanding the different gulfs helps you understand where you're lacking. And it's also that even if you fully close the gulf of specification and the gulf of generalization, if you don't close the gulf of comprehension, it doesn't it doesn't make sense. Let's describe what the gulfs are first if you don't mind somebody Shreya. Oh, no. I know I will have communicated this well if someone else can describe it. so I'll tell you how this figure came about. I thought about what are all of the problems that I face when trying to write LLM pipelines, and what are the different hats I have to wear when building good LLM pipelines. And one hat is my error analysis hat of I just have to like make sense of what's going on not just in the data, but also like not just in the LLM outputs, but also in the data itself. What are the different clusters of data? If you're familiar with data science, you're familiar with this problem.

The other hat that I wear is, you know, how to be a good communicator because I have to write prompts, right? I have to specify everything in excruciating detail to the LLM for it to understand. It has to be unambiguous. And this is very hard to do. And then the last thing is your traditional ML problem, right? Like I've written the pipeline, but does it generalize to my data? Does it generalize to other people's data? If I deploy it out in the wild, will it work?

and this is kind of This is what we call the gulf of generalization, which is does it You How do you improve How do you do task decomposition, fine-tuning, prompt optimization, all of these strategies that you have to invoke putting your ML hat on. So I would say, yeah, these are gulf of comprehension is your data science hat. Gulf of specification is your good communicator, good writer hat. And then generalization is your good at machine learning and software engineering. So And so can we go into a little bit more if you don't mind, like Okay, I don't I haven't memorized it just yet.

I really like this articulation also. So like gro- gulf of generalization is, for example, you have correctly specified everything in the LLM, and you know, you have a good prompt, but the LLM is still failing to do what you want. And that's like a gulf that you have to cross. you know, okay. And then gulf of specification is like is it just correct me if I'm wrong, that's when, you know, you haven't refined the prompt enough.

you haven't specified your intention correctly. You have to do that iterative cycle. What's What's comprehension again? Comprehension is making sense of the data. So also we We could change these names. I'm not tied to the names. But what we say when you say look at the data, the artifact of that is like some knowledge in your head of what your data is about and what your failure modes are about. that you I see. So you

have to have like comprehension of what you want to be able to specify it. Yes. Yeah, and that's what we say of, you know, there's this iterative cycle, right? Like you write a prompt, you look at the outputs, then you learn something about what's going on. That learning part is this gulf of comprehension.

One thing I really liked about this gulf and you might want to spend a little bit more time talking about it, like why you created this mental model. One anecdote that I really liked in the paper is every problem that you encounter, you have to build the bridges in you. I remember that phrase. b- and I really liked that. That helped like make it sticky for me, like this this water and these islands. you talk a little bit about like, "Okay, yeah, what made you want to create this mental model of like this cycle and these islands and stuff?"

Yeah, short answer, I'm HCI researcher or I'm a CS researcher who likes doing HCI. and I like to study how people iterate on pipelines, and I I really figured that these are the three hats. This goes back to like these are the three hats that you wear and you're interchanging between when you're kind of pi- developing pipelines. but yeah, that's nothing more to that. I mean, it's very nat- natural, Shreya, but I guess a lot of us have a lot I I can I I learned a lot from this. And I especially like that it's a tripod.

You can't just have two legs without one of them. Yeah. and and it helps me think a little bit better, so like when someone comes to me with a problem, okay, maybe it's Is it a problem that's not well enough? Or do you not fully understand your errors? Or are you just not thinking about the long tail? That's how I think about specification, comprehension, and generalization. Yes. I also do think that there's another gulf though, which is the gulf of model capability.

in the sense that you could be doing all of this right, but the just the model just cannot do this right now, the current generation of models. you see that sometimes, yeah. That's very interesting. I always I imagine that that is kind of part of this generalization gulf of like, okay, it's not working, but I've specified it so well. So, now I have to I have to wait for a better model, or I have to fine-tune models, or I need to do rag, or task decomposition, or whatever it is.

but it it's interesting that you say this because I I think a lot of people face this problem, myself included. Is you don't know when you've like hit the ceiling. That Yeah, it's hard. It's hard. Right? It's like, how do you know that it's just the model won't work for this kind of task, right? Like, maybe if you decompose the task better, or if you you fine-tune the model, maybe it'll work, right? And I think a lot of people Well, first, very few people get to that stage where they're able to push the ceiling.

But when you do get to that stage, and it sounds like you've been there, I think I'm curious if you feel like you have any tips or strategies for one, like knowing you're in that you've like pushed the boundary, and then two, what do you do? Do you do you wait for a new model? Do you just say, okay, I'm going to do something else? So, in just to speak more about the gulf of generalization, how I think about it is that, okay, your model works probably 50%, 80% of the time, but there may be a few edge cases that in the long tail they just haven't taken care of. And you have a great example in the book that talks about this, about the email name extraction.

That's how I think about it. So, the gulf of generalization is that there are a couple of edge cases that you just haven't taken care of. And in the long tail that's going to happen. Yeah. The gulf of model capability is you try and do something and it just cannot be done. So, maybe 2 years ago, if you tried to summarize a video transcript, you could try to do a lot of map map reduce hacky ways, but the quality is just not quite there. The tone is not quite there. It gets It gets It how do you say it? And you know, now this is not a problem. You can feed in the entire thing and Gemini will spit out the entire transcript for you.

and it can even do all kinds of modalities, right? But in the past Yeah, that's a relatively new thing, and it's really good now. Yeah. And and we take it all for granted, but if if you had to if you were asked to solve that problem in the past, you'd be like, banging your head against the wall, trying all these smart hierarchical map reduce techniques. And I know there's a lot of packages around that that do all that, but all of that is like obsolete for for good or for worse, in the sense that a lot of things are just simpler now. It's just all pushing it in the model.

but that's the gulf of model specification, whereby it doesn't work for more than 80%. It doesn't work more than 80% of the time. So, it's like That's how I think about that. I really like that distinction. I might steal that, quote you, put it back. That's what this is useful for. Yeah, so Oh, sorry, keep going. So, I just want to say that I love this. No matter what your editors tell you to do, do not cut this out.

This is the one of the things that I highlighted in green. I was this is novel, new to me, and I will apply it in when I help teams triage what the errors are. Oh, that's so great. cool. How about you had a question before, and we talked over you, or No. No, I'm fine. Okay, so that's good. Cool. so, the question that I wanted to get to, which I already forgot, so let me pull up my Google document again.

Yeah. Oh. Yes, so regarding the gulf of specification, I was going to go to the gulfs. what are some ways, concrete examples of where you've seen people get lost in the specification gulf? Going back to your statement earlier about how people just get lost in prompt writing. So, I I think there are two main failure modes. And then the first one I'll tell you that's the most common is like people learn about hear about this thing called future prompts, and they write a single example.

in older models, and to some extent in the current models as well, when you provide a single example, the model overly focuses on that example. It just gives too much that much attention to that. And sometimes, when you run a lot of samples, you will regurgitate just that sample. In a lot of times, it won't return a sample, but you'll return output that's very swayed by that sample. So, for example, imagine if you were recommending clothes for some reason, and you provide two examples, both female fashion, both high fashion. Now, when it gets to a point where where you're recommending clothes for males that's not high fashion, it the model just gets swayed too much. She learns too much from the in-context learning.

Interesting. And then people And and then people come to me like, why is this happening? It's like, have you tried removing that example? Remove that example, then it's better. But it's worse on the the reason why they added it. So, the reason So, that's what if you want to add instructions, first you try you try to add criteria that cover the examples. If you do want to add examples, you do need to add wider number of examples.

Like, essentially, your examples need to be representative, so that you do so that when you're trying to solve the gulf of specification, you don't accidentally open the gulf of generalization. Oh, that's such an interesting take. Yeah. So, okay, summarizing what you said, one is start out with specific criteria, and then two, it's very similar to traditional machine learning and data science, which is have a representative sample, like, take your data, actually create a training set out of it. Have a representative have a representative sample of that in your prompt.

because if you don't do this data science well, right, you're opening other cans of worms that you're going to try to fix that could have been fixed with your correct methodology in the first place. Yeah, so increasingly, I'm thinking of in-context learning just like regular machine learning. Your input samples need to be fairly representative. You definitely need a lot less now. You may not actually need entire sam- samples like a data frame samples. You can even write higher level criteria or pat- or patterns. It can do pattern matching fairly well, but it needs to be representative. So, now it doesn't mean that, okay, you have to cover the entire all the edge cases.

There will still be edge cases, but you cannot be overly specific. You have to put in care when you're writing this prompts and future examples. How How do you feel like you can iterate on this and make progress? I know this sounds like a very vague question, but often in this stage, right, people don't have emails in the specification stage. And you kind of have to make gut decisions on what examples to include, or what not to include. Can you talk a little bit about how you approach that?

Yeah, so I think there's I think there's two main splits. The one split is if you already have a product, okay, you just look at the data, try to figure out what kind of queries, what kind of actual input traffic will come in. Mhm. And then you look at that, you try to find out the main patterns, and then you write based on that. Now, if you don't have data, if you're building a completely new feature, let's say a chatbot for a website, you probably have to guess what kind of queries users will come users will come with.

and you know, sometimes it can be very basic. And if you have a good understanding of a product, you and you've got to understand your users, you probably will know what kind of queries will come in. And you may also need to come up with team queries. So, that's how I would do it. And maybe you just need like maybe five or six big buckets, maybe 10 or 20 questions from each of those. I I think that would go a very long way.

Mhm. Nice. Do you feel like there's ever such thing as too many examples? No. As a data scientist, I definitely don't think that. I I always I definitely do think that more data is always good. And you know, with LLMs, you can very easily analyze it, and it's just standard. I think there's also another example another problem with specification Mhm. is that, you know, you you give an example of how you write prompts, right? There's rules, etc.

Another problem I see is when users write the system prompt, and it's written in such a way that actually leads to a foot gun. So, for example, you may want your model to be very friendly. You may want your model to be trust to You You might say something like, you are a trusted advisor, right? Right. Now, what happens is that

the model may become too candid and frank, and perhaps even judgmental in the responses. Mhm.

Because you you have prompted it that you are trusted, I trust you, you should be open-minded with me. what happens be- because we try to have to push it in a certain direction, it should gets pushed in the wrong direction. In this case, the advice I've given, and the advice that the labs have given, is, you know, just keep it simple, don't add too much of this superfluous. Yeah. That was also interesting to me because it is very context-dependent and cultural, right? You might say something like, you're a trusted friend or advisor to somebody, and they if they know you, they might say, okay, this means like I should handle this conversation with care, I shouldn't say things that are mean. Exactly.

Whereas like in a completely different culture, you could be like, be open interpret that as being open and honest and direct. So, so let's let's go back to our fashion example again, right? You are a trusted fashion advisor. What do you think about this outfit? And then maybe the LLM might respond, Eugene, come on, how old are you? Why are you thinking of wearing this? You think you're still 20? So, something like that, right? You can imagine that in the long tail, things like that do happen.

so, you just do want to be careful about it. And of course the final gulf of specification is that because we keep trying to solve the gulf of specification, we try to make it more and more specific, we actually forget to refactor our prompts. And that is a lot of times when people come to me with prompts that are just not good. I just cut off I just cut it in half or like reduce the space and it just works again.

so I think like prompts are now like code. You do have to constantly refactor out functions. It's just like sometimes your prompt just returns many many different scores for different tasks. You maybe want to refactor them out and then you want to simplify that well. And because if you have if else and it works just as well, it saves you problems downstream. Totally. I like the saying of prompts as code because because that is what it is. But it's a little different, right? Because you can't statically make sense of it as well as you can make sense of code, right? When you read a Python function, you know exactly what the computer does if it's a small Python function. No questions asked.

But sometimes you just don't know what the LLM so I agree with that and recently I've been I've been I've been a little bit skeptical of agentic workflows and if you follow my account, you you you probably see this. but recently I've been trying it. I'm trying to push it. And you can write a single prompt like maybe two pages long and get the LLM to do the thing Yeah. for hours at a time and it will work.

Now tell me, is that not an ETL pipeline? Is that not an ETL pipeline for example of one? Just doing all the things doing web search and doing all the summarization everything. It is That is a That is a function where you just give it an input and it comes out output. Now it's not the standard kind of code that we see, but actually it works like that. maybe the most energy inefficient ETL pipeline in the world, but I do like the framing of that.

Yeah. Anyways, I know we're really banging everyone's head around this this gulf's three gulfs model. but maybe let's let's go into things that are more specific to how to go about evals. So one of the things that we Just

to linger a little bit on the gulf thing. Everyone loves the gulfs. Why why is it useful to know what gulf you're on? Like it's worth like just like frame let's like repeating that cuz my brain is like why these gulfs?

I know, but it's good for you to explain. I think we'll actually have different answers. My answer here is you it it helps to know what hat to wear, what tools to be having in your tool belt when you're going about solving a problem. So if you're in a situation where you recognize like, oh the model is not powerful enough. Right? Like you want to be able to switch modes to this and do research into like or think about like how to construct fine-tuning data sets for example. If you're in a situation where you don't know what is in your pipeline, you want to wear your data science hat or open up those tools and figure out how to move there.

that's what I think is most powerful. But I'm sure others Yeah, I agree with that. I think it helps you diagnose what the issue is. And if you're diagnosing the issue So let's say your prompt is great, but you think still think it's a prompting issue. But it's actually that you haven't looked at the data enough to So that's how that's how it helps you focus your energy. Yeah, I feel like the data island is ignored too much. No matter how much we repeat it constantly, that's the one that people don't think about. It's hard. It's really hard to look at data. People don't quite understand how to look at data. And sometimes when I plot a table, I I have all these metrics, people be like, how do you even read this table? Like maybe because I've just been looking at data for I I wish I could teach people how to look at data.

but I don't know. Well, we're going to get to that soon. I'll have my last comment on this. I guess this goes back to Hamel's question of like how this came about. I think that all of these pairs of islands are like of active areas of research and development all the time, but only application developers uniquely actually have to consider all three. Like I'll I'll tell you exactly like developer and data like that bridge, there's so much data tooling, there's so much data science frameworks everything to like there are whole industries around data science for this barring the LLM pipeline. Right?

Developer to LLM pipeline like this is all like HCI and UX in chat GPT and you know, how can we collaborate with LLMs? LLM pipeline and data, that gulf of generalization like the entire machine learning and AI community is focused on, you know, we have very well-defined tasks and how do we just make models solve them well. but when you're building an application, you got to do all three and there's no nothing around it and I think that's kind of what has made people really stuck for a really long time.

But anyways, okay. Enough of that. That's a really good explanation. Yeah, the gulf of generalization I feel like that's the one that foundation model providers are trying to solve mostly. and they're telling you as a application developer like don't put your boat around this cuz we're going to steamroll you. you know, but it is it is still useful for application developer to know cuz like things like fine-tuning you know, you might want to pay you need to pay attention to that. Yeah, exactly what Eugene said earlier like this model capability may not be there.

all right. So finally moving onwards, the next thing that we kind of really first introduced and this this, you know, is our first attempt at it. So Eugene, welcome all criticism and feedback here. I just want to say that this is the hardest thing that I had I think to me. No matter what, you have to fix this. Yeah, that's the biggest thing.

yeah. but my question for you was going to be I'm sure you've done this cycle over and over and over again at Amazon. Still probably doing it right now. So two questions here. Which box do you feel like you spend the most time in from analyze, measure, and improve? And the other thing is what are either sub boxes of these that are important to the elevate to their own box?

or some boxes to be aware of, you know, in error analysis. Maybe there's some big thing involved in there, but it's still clearly in error analysis. I think the hardest box is measurement. Hm. so so I'll talk about it. I think analysis is like you look at the data once. If you know what you're doing, you can sort of figure out what the key elements are. You can create a few hundred samples as your eval set. And that's one and done.

It's like good cap assist. I don't mean it's one and done. It's like the big chunk of that is done. And then improvement is like once you sort of have a good reward model slash auto evaluators, you can sort of improve on that and that's somewhat of a software engineering problem. Hm. What is really really really hard is how do you translate your evals into an automated black box that measures yourself against it? And a a big part of this is like, okay, everyone's using LLM validators right now. How do you calibrate your LLM validators to perform well on the evals?

It's really hard and I wish there was an easy way to do this. I haven't figured out how how to do this. Like the app I built, Align Eval, was trying to do this, but even then optimizing the prompt, trying to get it to where it is, it's still very challenging. And if this measurement is not good enough, Hm. directly, you won't be able to use it to optimize the generator. I think of the auto eval that you the LLM that the LLM judge is a validator, right? And then you can use it to optimize the generator. If this validator is just not strong enough or reliable enough, there's no point going to try to to optimize the generator.

I have a question for you, Eugene. and the reason why I say that this is the hardest thing and takes me the most time. I can get other people to do the analysis. I think PMs and subject domain experts are very good at that. I can get other people to improve the to do the improvement. Software engineers and other scientists are that very good at that. I haven't been able to find I I found it very challenging to try to figure out how to take the qualitative and convert it to quantitative.

Hamel, you had a question. Yeah. So when Shreya posed that question, I was thinking in my mind I feel like I spend most of my time with the in the analyze. I know Hamel's going to ask. And my question is, do you find that people can be effective in the analyze stage if they're not data scientists? And how do they learn that skill or what what do you find? I mean, you men- you mentioned it briefly already. I just want to That's a good question. I think they can be taught. I think there are some people if they feel a sense of ownership over the feature and they're able to put themselves in the shoes of the customer, they can try to understand this is bad, this is good. And you know, we have good methodologies to do this and you know, Shreya mentioned a bit a few of them like later in the chapter like, you know, binary only our preference selection. I I think those become reliable enough.

what I find that, okay, I could label some data in a day and I could get a couple hundred samples, but then I'll

spend a lot more time than a day trying to calibrate my auto my LLM evaluator to match your samples well. We got to send you chapter four. I'll do that after this. Yeah. Sorry, we didn't send you that. I didn't want to make you read like so much. that Yeah, going off of Hamel, I think Hamel, when you asked like you spend a lot of time in analyze, I I wonder if, you know, like big companies just like have the infrastructure and also the people to be able to follow these kinds of cycles well. Like you kind of already know who you can go to in your company for analysis. You have like maybe a role in mind for measure and and often, you know, people a lot of a lot of the people taking our course are not at big companies and maybe that is why they get stuck in analysis to some extent.

it was very interesting yesterday at the LangChain conference someone asked me what role or like who to hire to help them do evals. Like what are the skill sets they should have? And I found that it was like this weird thing of they should be a data scientist so they can do analysis but also they have to solve the problems that Eugene said. And that's not all data scientists. So It's actually very hard and I I've been through this exercise thousands of times. That sounds very data science-y though. Even the measure part. It sounds very data science-y, right? But I find that the analysis part, if you can put it in a spreadsheet and you can just tell people to say one or zero Yeah. or left or right you can actually decompose it. Yes, it takes a bit of time. It takes it's a muscle that the the team needs to learn and it's a muscle also writing SOPs.

Writing evaluation annotation criteria is a muscle. But I think once you build that muscle, you can do it. Like you can spin up a new task. within a week. So Now the the blue boxes Yeah, one I want to figure out. Like question I have is okay, I agree with you you can get people to label, you know, binary labels and go through spreadsheets, things like that. One thing I found that I took for granted is the skills involved in studying that data afterwards.

Like aggregating it, slicing dicing that data, whatever. You know, we have a lot of skills as data scientists to just do we think it's like breathing air or something very simple. But I found that people struggle with okay, analyzing it. Yeah, I agree. Okay. The second part of categorized failure modes, that's slightly more tricky that not everyone can do. As part of the analysis box. why don't we go into that? I think that we'll be skipping chapter two.

Which based on this conversation, honestly it's interesting that we immediately went to that and maybe it makes me wonder what the role of chapter two is. Like giving kind of foundations on LLMs to some extent. I'm curious how you felt while you were reading chapter two, whether you felt that this was useful. Like just for folks who've not read it, like it is basically like It's it's insane. You almost like it's like almost you read my mind. I actually have chapter sections 2.2.

Mhm. that in orange. Mhm. I felt and and sections 2.4, I like that orange. I felt like it was not going as punchy as I would like. I I I don't know, maybe like for beginners like some of these basics would be really helpful. But I feel like I just want to get into it. and this was sort of in the way. Mhm. maybe it could be an appendix. I I don't know. Or maybe it's like I'm not the right audience. but these were the two things I highlighted in orange.

Okay. Sorry. is great and this is why we haven't released the book yet cuz we're going to iterate on it. Yeah. And that's what she was also saying to students. Like asking right? I wonder if we should have this and then those are the two things that I like jumped at me. Yeah. I think you make a good point here. I think the audience that we had in mind here are like software engine literally the title. Engineers and technical PMs.

Probably they don't know I'm assuming that people don't know about LLMs. but I also agree with you. I think it's a lot of stuff, right? Like especially for somebody like you, you you know what LLMs are. You kind of you you have a lot of experience with them and therefore you know what you should expect and know what not to expect in ways that other people might not. So I think we can definitely like move parts of it to the appendix if not a lot. put reader notes to signpost like get out of here, don't think about this.

I agree. there is I guess we have a little bit of stuff around and this was I think we somewhere in here I cited you. Yeah, I think I think I had a sticky note. It's like, hey look mom, that's me. Yeah. yeah, at some point we kind of transition into how to think about evaluations, what types of metrics, and then also how to think about evaluations, what types of evaluations you want, how to think about getting feedback. I feel like this is a little bit ad hoc and I'm curious what your take was on how you were how you felt when you were reading it. Did you feel like maybe this would be better with a reordering or we should just kind of move stuff later?

Let me look at section 2.7, right? Mhm. Nothing actually jumped out at me. I do think that there's one section I highlighted in I mean if we are going into details here, the third paragraph I feel like the sentence other methods focus on fo- focus instead on relative compa- I think this is great. Mhm. I really like this emphasis on relative. a lot of Likert scales, a lot of grading it's just too subjective and it's not consistent. I think this relative this approach is actually going to help us solve that.

But there's another one which is that which summary is more factually consistent with the source document. I actually cannot disagree with this. I feel like this is a binary statement. It's either factually consistent or not. Mhm. in terms of relative, it could be like more things like tone. Like which tone do you prefer? so that that that was the two evaluations. Oh, I like that. Yeah. I think it's a good point. I think when I was writing factually consistent, I was thinking like which one fits better with the content. But that's that's not factually consistent.

Like which one is more concise or which one is more comprehensive? They they there's some things that just hard to draw line on. Yeah. Something that's clearly That makes sense. Thank you for that. okay. So all of this I'm realizing let's we'll we'll change that up. But let's get to error analysis in the last kind of 20-ish minutes. so I'll start off by giving a very brief high-level overview of error analysis section. this is this is from Anthropic by the way. This is not from me. So this is kind of what inspired me in the first place. I would say this is like a year ago where I saw this and I was like, oh it's nice to think about this workflow of like what I should be doing as a human being building LLM applications. but I also did not know what this box meant. So that's motivation for this. And I'll I'll go to the first figure where I think where we think about like what are the steps, you know, that you might do when doing error analysis.

here and first starts with just coming up with an initial data set. So I'm really excited to ask you this question

because you are at a company in which you probably have a lot of data. and when we wrote this, we wrote this for people who might not actually have data. So for example, generating synthetic data and here's an approach to coming up with synthetic data that might be better than like simply asking ChatGPT or like asking some random person. Hamel goes on this big rant all the time of how people outsource everything. Don't outsource your data generation.

but we we come up with a little bit more of a systematic way to do this. so I'm curious what your thoughts were when reading this. Did this first is it something at all that you agree with at the high level? Then second is do you guys do this at Amazon or a version of this? What is that and what are we missing? So firstly, all opinions are my own. I'm not speaking on behalf but that's the that's the first thing.

So I should have not said that. Like so what do you think people should be doing? Forget Amazon. I think this is a good idea. I especially like we have a statement in the next page. whereby you you shared a approach where you do a tuple, right? So you have a statement whereby if we let if we let the LLM to generate full queries in one step, it tends to produce repetitive template patterns. Fully agree. what I found is that you kind of need human judgment to say this is the outcome I want.

Or let's just say you are generating again fashion recommendations and you know, you if you ask it to generate fashion recommendations given the same persona and given the same thing, it's it's going to be generating very same things. But now you may want to push it in a certain direction. Okay, I want fashion recommendations that I want you to generate queries for edgy fashion recommendations, cheap fashion recommendations. Well, I I don't know. Like all the different edge cases that you know because you have worked in the business and you know what kind of queries your customers are asking.

And you nudge the LLM in that direction, it will be more useful when you generate the queries. before I even read this, I've actually done a bit of this. Be- because I just wasn't getting the kind of queries I want. It was not as rich as I want. And this was the this was the way to do it. It's almost like instead of generating chain of thought you generate rationale where you give it this is the final output I'm supposed to expect. Generate the rationale and then you feed that rationale into your few-shot prompt.

Mhm. I think this is the same thing. This is what I what I achieved and then we done. So I think it works. I think it's a good idea. Interesting. Do you find that it's hard to think about So, I I told this I stole this example of the real estate chatbot or whatever I called it, real estate agent, from Hamel cuz Hamel has Also, you should read Hamel's blog post about how he kind of worked on this.

Real actual use case here. But as someone who didn't at all, I I had to go through this exercise myself. Basically, cuz I'd never worked on it. And I actually found it pretty hard to think about like I'm not even a real estate agent. Like what What the heck What are the dimensions of a query could be relevant. and I looked at Hamel's blog post and I saw some example queries there and I was like, "Okay, I can break that down into like things that, you know, real estate agents want to do.

who they're working with. And And there is more, but then like this this scenario Scenario is not a good term here, but it was something around query ambiguity. and this was not So, this is just something I've seen on a regular basis. Like a lot of people try to submit ambiguous things to agents and you always want to know how well the agent handles ambiguity. But yeah, I was surprised at how hard I I found this exercise to be. and I I used ChatGPT a little bit to to think about like what these dimensions might be and it gave me like 15 dimensions.

And I was like, "Okay, I don't know if these are the right dimensions." So, I'm curious both of you actually. How do you go about coming up with dimensions? What parts of it are hard? Is it okay for it to be hard or am I just weird? No, it is hard. It's very like specific to each problem. It's like the golf. So, like you build a bridge each time. and these dimensions are are different each time. When I was writing the blog post, I was trying to make it concrete and say, "Oh, these are the dimensions I used that this time, but consider your dimensions, whatever they are." you know, it's not always features, personas, scenarios.

but it's more like grounded in again the data analysis. Like the error analysis. Like, "Hey, what what dimensions do I really need to vary the most? And what what failures do I want to elicit with this synthetic data generation?" And I have that in mind. Like, "Okay, I'm going to elicit this kind of error." Because I want this to pop up. So, like I make sure like that's in the you know, this like scenario somehow. Edge case generation.

Fortunately, software engineers are very good at this. It's not like a specific to data science, but Yeah, I would echo what Hamel said. It is hard. but that's a little bit of understanding of the business helps a lot. So, again, the fashion example. Yeah. Let's say you try to elicit queries, right? and you may know what kind of queries your users send to your fashion advisor. They may be sending query with I have I actually don't know what fashion pieces look like. I'm just going to make it up. I have this Air Jordan 2027 shoe. Find me a pair of jeans that matches it. So, now the the query is very specific to piece of article. Or I'm going out for a nice summer day. Or I'm going to Hawaii.

so, it does a scenario. Or there may be, "Hey, find me things of this specific brand." So, now you have three things, right? Like there's there's the item name, there's a brand, and then there's the scenario. So, based on this, you can generate queries that meet this scenario. I I don't know if I'm using the right the the word scenario correctly. But that's how I think about generating queries. not Scenario is not the right term for me to do. Like the the Yeah, the triple or the basically the tuple of dimensions.

what you said about the fashion example reminded me of my footnote here. and this is definitely true of search. This is a long footnote. But basically, in search, there's only very few query types. Or the taxonomy is small. and I don't know if this is true with agents or not. And I'm curious what your take is there. I think that I think it's the Pareto rule.

you'll find the Pareto rule. 20% of queries address 80% of what users want to do. and the same thing for search. So, I I think that will happen as well. There's going to be a long tail, definitely. And of course, you may choose to choose to address or not address that long tail. It's up to you. But I think yeah, you can you can get by power with just 20%. Yeah. Especially, I'm thinking here with agents. Like perhaps even a longer tail because you could

put like essays into these ChatGPTs in ways that it's harder to put an essay like Google search.

Anyways, that was an aside. But so, moving on from kind of generating synthetic data and synthetic queries in a more kind of interesting way. So, the next part of this kind of talks about how to even like go about qualitatively coming up with the failure modes. and I drew a lot of inspiration from grounded theory from social science here. They do have names here for these kinds of what's called coding. It's not our engineering coding. but it is like writing on the data. And I tried to cite some stuff for the social science geeks in the room.

but I'm curious if this is something that you guys follow to some extent or you do. and what's similar, what's not similar. So, for for the audience, I talk about a two-pass approach here. First, you go through every trace and kind of write open notes and label them. And then the second pass, you try to merge these first-pass annotations together to come up with different failure modes. So, I'll hand it off to you, Eugene. What do you think? And then Hamel.

Yeah, I think it makes sense. But I also want to share how this can go not so well. Mm. So, let's talk about again our fashion example. You may have fashion experts that may say that Okay, you know what? I actually can't do fashion. I'm going to change a little bit. I wish I could be using Let's talk about translation example. So, imagine that you could have errors where the idiom was translated incorrectly. Or the text is overly literal. Or the text is awkward phrasing.

Now, to an expert, this might seem like three things. But you may try to get an LLM evaluator to to distinguish between these three things and you may never succeed. To a layman, to me, I actually don't I can't I can't really tell the difference as well. Like idiom over literal translation and awkward phrasing. Mm. Then why don't just merge them? So, now this is where you actually have to go against your expert expertise, your expert judgment. You're saying you know, these are these these are three things by this definition.

Whatever whatever. Then there are instances where the the category is so big. Such as the category could just be a category called accuracy. Now, because the category is so big and then you try to get an LLM evaluator to catch this, you actually lose a lot of false positives. Right? Because the category is so big, you have to write very broad instructions. Now, if you want to reduce your false positives, it may help to split it up.

It could be like, "Hey, you know, you got the subject wrong. It's male or not female." And you know, this happens in certain languages where nouns have subjects like Spanish. Or the category could be like, "Oh, no, you got the formality wrong." And this happens in German where there's there's a different level of formality and switching the formality in the middle is very jarring for German speakers, as I come to learn. Or you may have gotten the nickname wrong.

Interesting. is not just accuracy in general. And you know, when you just say, "Hey, evaluate this translation on accuracy," the model will just evaluate grammar and spelling and everything. And that's what too many false positives. So, I I I feel it's an iterative process. Do you feel like this is I mean, you mentioned automated strategies here. Like, do you think that this is a problem that humans have when they are going through and

looking at their data? Or tell us more about that.

I think experts don't have this problem. I think laymen have this problem because we just not trained to that extent. Like I I can't tell the diff Honestly, when I look at the data myself sometimes, I I struggle to understand the expert's point of view. And then the then comes the question, right? Are we building this for the experts or are we building this for the general customer? Interesting. also it's hard you need to put yourself in the shoes of the general customer and how do you think about this?

Interesting. So, if I were to rephrase what you said, correct me if I'm wrong, it's that if you're not an expert and you're looking at this, you don't realize the nuances of how each trace actually should behave differently. And you try to evaluate them all in the same way, maybe. so, translation was the example that you gave with like, you know, German might expect something different than another language. and then you do that and you actually get bad evals. That was interesting.

I wonder if Yeah, the LLM is just this kind of perform well. Yeah, it's not necessarily automated thing. I think in this chapter, we can be more explicit about it. It's like this is not an automated chapter. This is like come up with failure modes. Don't like do the annotation yourself. And I think Once you we should be explicit that this is not an automated process right now. And then two, that you need to be an expert or call on an expert to do this.

for some of the reasons that you mentioned. I think that's good feedback. Hamel Yeah. I didn't interrupt you. No, no, I was just taking notes. I can also edit this as well. But yeah, I agree. I didn't have anything else to add. No, you tell people to look at their data all the time. Eugene, I have a question for you. Hamel, I actually bought a hat. So, where did you get this hat? Can we share the link? I need to buy this hat. Yeah, I need to buy this hat.

I made it. You have a I made this hat. online store or something? I just went I just went on Amazon and searched customized hats and just pick whatever that has the font I like. There's a thing Amazon you can customize your own hat? I'm going to buy it and I will ship it to you. Yeah. Cuz I bought a hat just for you. Okay, I'll buy one and ship it to Shreya then. We'll do the circle of pay forward. Buy one ship it to him.

Oh, wow, you can customize it. Can you customize it in the Amazon interface itself? You can. You can. It's I I've learned it's quite advanced. You can just type whatever you want in a text box and they'll print it for you. You can even upload images. I didn't bother. I just typed in the text box. Oh, interesting. That's how we astray we interrupted you. No, no, I want the hat. I look horrible in baseball caps, but I'll get it anyways. So, I'll take one for the team.

I don't even remember what I was going to say earlier. Oh, yeah, yeah. My question was do you when you're doing your first pass of error analysis, Eugene, like it's interesting that you thought of using automated evaluators first because that's something I've literally never thought to do. And like maybe that's because Hamel always preaches like No, I don't I don't do it. Like I make the people It's like, you know, Yeah. like, you know, I'm forcing you to look at the data. But yeah. But maybe there's something there to like Eugene, what kinds of metrics do you think you like are good for like automated evaluators in the first place, if at all? Or like why do

you go to that?

So, I want to clarify myself. when I look at the data the first pass, I do it manually just to get a sense. Mhm. But then once you come to the failure modes, and then we try it with the LLM judge LLM evaluator, we may find that the LLM evaluator is just not able to get it. Mhm. That's where we may need to revisit our failure modes. So, either merge them or split them. Mhm. And then pass it to the LLM evaluator again. So, it's a it's a cycle, right?

That's why I like the a analyze, measure, improve. Interesting. after you analyze it, you need to figure out if you can measure it well. If you can't measure it well, okay, back to analysis. Yep. So, there's there's an inner loop there. and the that that is the that is the loop that is the hard the hard loop, at least for me. Where I need to revisit the tools that I Some of the tools that I really enjoy are ones that Shreya has come up with in this area.

EvalGen is one. Have you It's now out. It's now out on Change Forge. Change Forge, right? Yeah. Yeah. So, it has this like workflow, very innovative human computer interface workflow where it's like you annotate and you build the judge and you iterate on it all in one integrated flow. Yep. I think that's a great idea. Which I think you know, which I wish was a commercial prod project. Someone needs to So, startup waiting to happen. It's hard to do hard I mean, took a year to merge domain, but but I think also from like How about this Shreya, you'll be CEO Hamel you'll be CEO right now?

No, no, no. I'm literally teaching a course and a grad student. Yeah. I think I think doing all of them the same time is the dream. And we there's actually a chapter in here, Eugene, I can also send to you on like how to build good interfaces for human review. Not in the first three, but anyways. Although we only have 5 minutes and I don't want to keep people on much longer. So, I might move towards some of the latter questions that I had and I think there's one more big one. So, I want to say I was going to wear this during the Finish.

Even though I I I don't really look good in this, but okay. I think I'll get t-shirts and hoodies and stuff. I'll figure it out. I'll up the ante on this situation. It'll be like a bucket hat or something. Yeah, the the reason why I say this is because Nick White on the YouTube channel, you should just have a flag or a sign in your background. Makes sense. And that's why I figured I'd just keep it keep it on.

Yeah. Okay, so the last question I'll ask that's like relevant to the book is around So, I mentioned we did one pass of just writing your failure modes and annotating and then another pass of trying to group them together, merge them, or refine them in a bit. And one thing that we mentioned here is actually using LLMs to synthesize failure modes out of your open-ended annotations. this is something Hamel and I have discussed about how we like to do this and it actually works pretty well.

but we're curious if you do it or if honestly if anyone else does it. or you know, how do you do this merging process? whereby you say you take all the error the failure modes and try to get it to cluster and do it? Yeah, basically. The LLM is a clusterer. Yeah, that's what I would do as well. I would take Yeah, I would just copy the errors and ex- the human explanations in Excel and then just paste it into some LLM and ask it to synthesize it.

I can see. I mean, it's worked for me. I think there's one key parts around like when it does come up with the failure modes, then I have to go back into my spreadsheet and like figure out, you know, what failure mode does it map to so then I can make Hamel's favorite pivot table. He loves pivot tables. or it's group by have clients that like this so much that cuz I one of the things that we talk about is make your own annotation tools.

Mhm. And clients like make you know, when they make their own annotation tools, they actually build this in now to the annotation tool where it auto groups and classifies failure modes. Wow. I Yeah, I have a note here that says that I think it really makes sense and like everything I treat everything I do I treat AI as a first pass draft. It's like everything I just Well, not when responding to you guys like or like, you know, just talking to my wife.

I don't I don't pass that through ChatGPT, but if you put it through AI, it will come up with a very good draft. It will come up with very good clusters. And now to be clear, there's always 5 to 10% manual tweaking you have to do yeah at the end. Or maybe as much as 50%, but it always saves you time and it almost always finds me patterns that I may have missed. so, definitely I will always do this.

I love that quote. AI is a very good first pass. And that's it. Yeah, I I feel the same way about using ChatGPT to do things. I I find that my workflow for this error analysis, this chapter three, is a very fragmented I know we espouse a lot like building good interfaces, manual custom interfaces, but sometimes I just export to spreadsheet and just do stuff in spreadsheet and then I will copy and put that in ChatGPT, it'll come up with failure modes and then I like go back and create a new column in my spreadsheet, manually add things.

group things ChatGPT can output CSV file pretty well. Yeah, I'm so bad at it. I don't know why. Maybe I don't have the like I'm using like oh whatever pro. I'm so bad at like prompting it to do anything with CSVs. Mhm. Interesting. Depends on how many rows. I've gotten it to return like 50 rows or 100 rows. Not ChatGPT or us like Claude. Claude, okay. Maybe it's the wrong LLM. I don't know. I also feel very impatient sometimes when it's like thinking about its code that it's writing and then it's like doing data frame.head and then seeing all these tokens come out.

Anyways, all right. So, 1 minute remaining. So, why don't I conclude I want to say that I really enjoy section 3.6. I wish there was some way I could highlight this. Oh. the thing is I wish I could put it in the front of the chapter, but most people wouldn't understand it. You had to You had to pay the effort of the previous sections to understand this very well, but this is a very good section. I especially like paragraphs two and three. Mhm.

Actually, I I like the entire thing. Skipping open coding, use of like the scales, treating annotation treating your SOP as a fixed criteria, those are the most common pitfalls. those are very common pitfalls. So, yeah. This is a Hamel section. He's like, "What are the common pitfalls?" I I always love working with Hamel. It's it's like, "I'll write something here like this made no sense." Like, "Make it clear, communicate better." And we you have him to out that being slightly dumb is helpful.

No, it's not slightly dumb. So, yes. Very good communicator. Anyways, it is 6:31. So, maybe any final thoughts?

what are you excited for in Evals, Eugene? What do you think's up and coming? I feel like the the field is stagnating. I'm excited for this to come out. I think there's going to be some new and interesting stuff here, but generally, what's exciting to you? What I'm thinking about right now is that, you know, you have that loop, right?

Analyze, measure, improve loop. Mhm. I wonder how much you could make that loop automated. Mhm. I think there are two loops in this. The first loop is the analyze and measure loop whereby how can you get your measurement to align with your analysis? And that's what I mean evaluators are supposed to do. Yep. But once you do that, and maybe this LLM evaluator is a little bit lossy, how can you get that to move upstream? Instead of catching the defects, improve the upstream generator. Mhm.

So, you can imagine this like maybe back propagating from your initial eval set to your measurement to your improvements. I I don't know how to think about that yet, but I do know it's very essential. would love to chat with you on that. I mean, maybe we should work on a paper already see the beginnings of it with, you know, with Align Eval, with Doc ETO, with Eval Gen. They all have like are tightening this loop.

And those ideas are not Some of them have make their way into products, like we saw some, you know, announcements from LangChain yesterday or day before. But, I'm you know, it's still kind of hidden away and, you know, we none of us here in the business of trying to create SaaS tools or anything. But, yeah, I agree with you. There's a lot of opportunity to to make this not completely automated, but put human in the loop in the right way to make it a lot more like a lot more automated.

I think this is the exciting problem of the decade. Like, how do you automate this in a good way, right? Like, not in a silly way that like assumes no human involvement or another silly way that assumes too much manual human involvement. And also maybe to just play devil's advocate against myself, right? Imagine you have a eval dataset cuz maybe 400 samples, and there are 400 samples. Instead of going from the analysis to the measurement and then to improvement, why not just shortcut it? Why not just put your eval dataset in the prompt?

With prefix caching and prompt getting cheaper, that's a 400 shot learning. Maybe that's good enough. Maybe I'm overthinking this, right? We have to do this loop, but just do the simple thing. I don't know. I don't think you're overthinking it. I think you can't get to evals without doing analysis. That's the thing. yeah, do the analysis and just pop it into improvement. You don't even have to measure it, maybe. Mhm. Interesting. Yeah, maybe. Just make the analysis so delightful and so frictionless that you want to do it and you're like and you see the payoff so fast that you just are motivated to just keep doing analysis cuz you And and just make every analysis a new few-shot a new few-shot prompt. Mhm. And then you fix the improvement.

Maybe I I don't know. Align Eval V2 is what I'm hearing. All right, so when are we starting the Eval SaaS together? I mean, Can I be an advisor and a silent partner? I I'll I'll invest. It's like that meme where everyone's pointing at each other. I It's No, it's like that meme where there's one person digging, Hamel's digging, and then we're all standing around him. I'm going to No, I'm not doing I'm going to I have to work harder at convincing

y'all.

Anyways, we're well over time. so, thanks everyone Wait, wait, before you go, can you talk a little bit about this book, Shreya? Like, will it come out one day, somewhere, where is it, where how all that stuff? So, you want to read it soon, take the course. You'll get like you'll get each section at each lec- lecture basically, and you'll have the whole thing by the end of V1. we are we don't know who will publish it with yet. We're working on figuring that out, but it will be a book that you can also buy later, but I wouldn't expect it until like end of summer, maybe.

Okay, and if people want to take the course, there's a 35% Yes. Discount code in the Luma Sorry for not selling. Yes. No, no, I mean, it's fine. So, I'm just letting you know in case it's helpful. or just find just look through Twitter pi- look through Twitter on any of us three and you'll find discount codes everywhere. Yeah, it starts next week, so really excited. I think it's going to be super fun. you'll learn a lot. Here's I'll I know now which chapters to send Eugene, so continue with human review. Just send the whole thing. Send the whole thing.

Yeah, okay. Yeah. Send the whole thing. Great. All right. Thank you, everyone. See you, folks. See you, folks. So, Hamel, you stop the live stream and we'll stay here.