SUPER DATASCIENCE
MAKING THE COMPLEX SIMPLE

SDS PODCAST EPISODE 920: IN CASE YOU MISSED IT IN AUGUST 2025

Jon Krohn: 00:00 This is episode number 920 our, In Case You Missed It in August episode.

00:03 Welcome back to the SuperDataScience Podcast. I am your host, Jon Krohn. This is an in case you missed it, episode that highlights the best parts of conversations we had on the show over the past month. We head first to episode number 913 where Julien Launay , formerly of Hugging Face Now CEO of the wildly successful startup Adaptive ml Julien talks to us about recent shifts in attention from pre-training of LLMs to post-training. To date engineers have tended to concentrate on pre-training models and getting the right quality of data from the relevant sources. I wanted to know why Julien now considers post-training to be even more important.

00:45 Before we get too much into Adaptive Your Company, I'd love for you to talk about based on your rich experience at Hugging Face, also at a company called Light On that we'll talk a lot about more later in the episode. Through that experience, you have tons of experience in creating LLMs that are useful for real life and at the biggest scale that LLMs come. So I'd love for you to start off by providing us with an overview of the steps involved in creating an LLM like pre-training and reinforcement learning.

Julien Launay: 01:16 Yeah, I am. It's a very timely question as well, given that I think the steps are blending a bit these days. So take everything that I say with a train of, so there's always nuance in this, but very broadly speaking, the way that historically large language models have been kind of approached. First is through a pre-training phase, which is the bulk historically of where the computer has been spent. Pre-training is during pre-training. We essentially collect data from all over the web, pretty much every books, every paper, pretty much nearly at the scale of modern pre-training, nearly every text in existence. I think it sounds very grandiose, but it's not far from being true. And even nowadays, images, videos, and all of this, and essentially the model is trained to very grossly predict the next world, predict the next token. This is a step that is built to be scalable, to run at scale, that are essentially everything we have ever produced on tens or hundreds of thousands of GPUs these days.

02:14 But pre-training is only your first step because immediately after pre-training models are actually a bit weldy. If you take really pure, pure pre-training and you try your model immediately after, it's not going to be very interactive with you. It's not going to be chatty, it's not going to answer your questions necessarily in the way that you expect. I think a failure mode that we used to see a lot immediately after returning is, let's say I ask a model a question and instead of answering the questions, the model will come up with 10 more questions that are similar.

02:41 And so reason why is because in its pre-training data, this is a very likely to have a list of question asked to have the answer following the question. And this led to the development of second phase in model training, which is called post-training. And the idea of post-training is to on own, in sharpen the model to really fit how it's going to be used, which typically means making it a good chat assistant or something like that. And the methods that you use during post training typically differ. I mean, strictly speaking, you could do post training in the same way you do, but with just data that is specialized, maybe just only transcripts of chats and continue doing pre-training on transcripts of chats only, and you would defacto be doing a post training towards the chat model. But very often people like the big success of post-training has been the use of reinforcement learning. So essentially enabling models to learn not from an explicit demonstration of what they should be doing, which is what supervised for engineering and what pre-training are, but instead from a feedback about how are they doing? So the model generates an answer, and then from a human from another model or from many different possibilities, the model gets a feedback of like, this is good, this is bad. And just based on this positive or negative signal, the model learns to improve.

Jon Krohn: 03:57 So this is the experience that a lot of us will have had in chat g PT where there's a thumbs up or a thumbs down that you can click after you get a response, and that can then be used as a training data for this post-training phase, and that'd be reinforcement learning from human feedback, RLHF

Julien Launay: 04:12 From a very high level point of view. This is an example of the sort of data you could be leveraging two power of this phase of post-training. I think what's really interesting is right now I'm giving a description where pre-training and post-training are very separate things, so reality is much less so these days first because now pre-training is very dynamic where you shift the data distribution. So you might start with the lower quality data, the more bulk data, and as you advance through steps of pre-training, you will focus more on higher quality data, maybe more code, more mathematics, more. It could be many more chat data, more of the higher stuff that you consider high quality. I put high quality in quotes because the definition of quality is a more subject that we could spend hard on and post-training itself. Even now people are starting to do reinforcement learning during the pre-training step or starting at some point where they start to incorporate mixed blend the two. It used to be that post training was a much smaller spend than pre-training. Most of the money used to go to pre-training and to the millions, tens of million, hundreds of millions of dollars used to go there. But now if you look at recent papers like Kimi or even Portfolio paper, but more something that mentioned, which is that they spent nearly as much on post-training as pre training. So there's massive scaling up of this post-training phase,

Jon Krohn: 05:35 Whether it's in pre-training or post-training of LLMs. In this next clip from episode number 915, I asked Michelle Yi about how to iron out LLM issues before the AI agents that depend on them make less than desirable decisions on our behalf. And so another really interesting, in fact, I think it's one of the most surprising and interesting research reports that I've ever seen. I covered it in detail in episode 9 0 8, which aired recently, and it's all about age agentic misalignment research from Anthropic where they found that 95 to 96% of the time for their own leading models, and it varied a little bit. Some of the leading models were as little as 80% of the time they would resort to things like blackmailing. So they were put in this simulated corporate environment with a bunch of corporate data. And so you have an age agentic framework calling these LLMs using the LLMs as their brainpower to be doing tasks.

06:39 And all of the leading AI models between 80 to 96% of the time, and a lot of 'em are 95 to 96% of the time, they would resort to things like blackmailing people and they'd dig up if they found out that there was going to be an update, a software update overnight, and they would no longer exist the next day. They're not conscious as far as we know, but just because of, I don't know, movie plots or whatever is in all the training data, the pre-training data probably that these LMS are trained on, they get the sense that they shouldn't want the thing that the next token that gets output is, I don't want to be shut down. And by the way, I found these emails that you're having an affair and if you do shut me down, this email will go out to your colleagues and your wife.

Michelle Yi:	07:34	Yeah, this kind of churns a few different thoughts on my side. One is we've been, agents are obviously the main stage of pretty much 90% of AI conversations right now. I'm sure you're tired of hearing about it at some point as well. And there's probably very few, I would say, scenarios where the agents are actually being very effective and useful in production. I think there's probably very few organizations that have this that mature, and so a lot of
Jon Krohn:	08:08	Call centers a good use case
Michelle Yi:	08:12	Research in theory. Yeah.
Jon Krohn:	08:13	Yeah.
Michelle Yi:	08:14	But how many people are actually using,
Jon Krohn:	08:18	Yeah, I guess it's hard to know. I mean, it is an early technology for sure. Sorry, I'm being defensive about this because this is, my consultancy is specialized in bringing things like solutions like this into enterprises, but it is early days, and I guess even from that perspective, I can answer your question that very, very few organizations are actually doing it, which means it's a great time to be.
Michelle Yi:	08:42	Exactly. And this is why they need specialists who actually know how to design age agentic systems in a proper way because I think so many people get lost in the pitfalls. They've been really focused on developing the best single agent, let's say the best suite, the best Devon or the best SRE engineer single agents. But when you start getting into collective systems and groups of agents and this decision making, okay, now I need to blackmail John too. And so I'm going to tell this other subagent that's the research agent and I'm the manager agent to go tell John that he needs to ignore the latest software updates or the latest research in alignment so that I can continue to survive. This is why there's a deeper level of

research and thinking and expertise that's needed to design these effectively. Yeah.

Jon Krohn:	09:37	Do you feel confident as somebody who's so interested in trustworthy ai, going to conferences like Black Hat, DEFCON, this being a lot of what you talk about, research about, do you feel confident that there's enough attention on it that we'll figure it out long term?
Michelle Yi:	09:52	You mean the trustworthy ai, trustworthy in
Jon Krohn:	09:54	General. Well, everything's going to be okay long term and we don't, we're not going to be overrun, do I have to use the word Skynet
Michelle Yi:	10:02	Here? Yeah, yeah. Well definitely we all get the reference, but yeah, I do think at the end of the day, the systems are out there, people are using them. That's sort of what's the English saying? The cat is out of the box.
Jon Krohn:	10:19	What is this? The cat is out of the bag. Yeah, it's always, it's a weird image even as a kid to think about why, who put it in the bag to begin with? Who was the sick person?
Michelle Yi:	10:32	Thank you for understanding my conflict with English as a fourth language. I also, I don't understand these idioms, but the cat is out of the bag from whoever put that in there. Maybe it was an agent. Exactly.
Jon Krohn:	10:46	Misaligned agent.
Michelle Yi:	10:47	Yeah, exactly. So I mean, I said give it a bath or feed it. I don't know what it was doing. How long was it in the bag? But oh my God, I don't get English saying sometimes, but so it's out there. Is there going to be enough investment in solving trustworthy ai? Questionable, but I do think it's a, it's not too late. B, we should figure it out. And I know you've had other conversations with guests around the

policy side of it, but on the technical side, I think there's a lot we can do as well. How do we detect or invest in techniques that detect when data is poisoned, when there are malicious actors or how to prevent hallucinations and some of the investments? So for example, world models, there's a ton of investment in world models because for many reasons, but one of the great applications of world models is actually that hey, we can self simulate if something bad happens to prevent essentially a hallucination. So if you told someone to walk off a 20 story building or something like this as part of the conversation, the model with a world model would be able to understand like, wait, this is a pretty bad scenario.

Jon Krohn: 12:05

It's a scary thought. So this phase in our development of AI will be critical to ensure that we can stay ahead of our models. And here's a great moment to introduce the new super data science AI engineering bootcamp, which trains participants in all the steps toward building AI systems We can trust in episode number 917, the founder and original host of this podcast, Kirill Eremenko walks me through the full eight weeks of his bootcamp. I can't wait to learn what I need to know as I become an AI engineer. Let's start with week one is probably the best place

Kirill Eremenko: 12:38

To start. Well, let's do a quick overview for the background, what kind of prerequisites they are for somebody who wants to follow this kind of curriculum. Bootcamp is designed to take people from a intermediate, high, intermediate level to advanced or starting advanced or medium advanced depending on where you are now. So it's quite a tight range you have to be in to do a bootcamp like this. And the outcome is roughly advanced level, maybe advanced plus, and the prerequisites that we expected and we asked our participants to, you have to already know Python, so there's no learning python in this bootcamp. You have to already know the usual things like a bit of PyTorch, a little bit of psyche, learn typical work with pandas, even though we don't work a lot of pandas, you have to be confident of those things. In addition to Python, you also need to know LLM calls like API calls for LLMs.

13:38 They're not difficult. We recommended some participants that didn't know those to get an overview before the bootcamp because we don't want to spend too much time understanding what an API call is. That's the typical way of calling, whether it's Chad, GPT, also open ai, LLMs or philanthropic or GR or whatever. And also some cloud experience because the way, and the cool thing about AI engineering is that in our view, to be a successful and effective AI engineer, you need to combine two things. One is the science of ai. How do you build AI that does the job? How do you build the proof of concept to solve the business problem that your business needs to solve? By the way, pre-phase to all of this, the goal of AI in this context is to solve business problems, is to add value to businesses, not just ai for the sake of ai.

14:32 So first of all, it's like how do you build a proof of concept that will solve the problem effectively? And that's the science of which LLMs do you use? How do you combine them? How do you augment them with rag? How do you add things to them? How do you use agents? Do you not use agents and things like, so that's the first four weeks and the second four weeks, weeks five to eight, that is deployment. And that's a second imperative component of a successful and effective AI engineer is to be able to deploy systems into real world environments, or at least to understand what the deployment takes. So how do you now take that proof of concept ai, which is like a Jupyter Notebook and how do you put it into a cloud environment? The one we used for the bootcamp is AWS. It can be Azure, it can be GCP, it can be any other environment, your own servers.

	15:20	But you need to understand how do you take that POC and put it into real world environment? How do you make it secure? How do you make it efficient? How do you make it cost effective? How do you make it reliable? How do you make it scalable? Those are all important constraints that don't exist in the world of proof of concept, but they are critical for business, real world business systems because what if you have one user using it and then the next day you have a thousand, the next day you have 10,000 using? It has to be scalable, it has to be secure, it has to be reliable, it has to be cost effective. A lot of people don't think about that, but do you deploy it on serverless architecture? Do you use a server? Why do you choose one or the other? What's your trade off of a speed of responsiveness to latency depending on the business application? And so the weeks five to eight focus exactly on that. And the way we described this at the start of the bootcamp was we have two instructors. One, the first instructor is your good friend, ed Donner. He was fantastic. So we described it as Ed explains to you what is possible with AI creates this huge bubble of dreams. And then in weeks five to eight, our second extract to Sam, who I've been working with for over a year now,
Jon Krohn:	16:29	What's Sam's
Kirill Eremenko:	16:29	Full name? Sam Baston. He's an expert in cloud. He's an expert in a WS he's been doing for 15 years, and most recently the past two years he's been doing specifically LLM deployments, LLM and AI deployments. And then the second half of the bootcamp, Sam comes in and shrinks your dreams back to reality because not everything that's possible in a proof of concept is going to be possible in a real world deployment
Jon Krohn:	16:54	Rule number one of developing an AI model should always be that the initial idea is grounded in reality and

can solve a current problem. In episode number 911, Akshay Agrawal talks me through his solution marmo, which opens up notebooks across the board in a company from data engineers all the way through to CEOs. You've talked about a vision where you go from, and as we've been talking about this whole episode where you go from these kind of error prone JSON based scratch pads that Jupyter Notebooks are into a full stack developer platform where exploration and deployment converge just as in having a data app there ready to go instantly. So when tools collapse the boundary between notebook and application, that seems to change, not just workflows, but also how roles work. As this line blurs, what of new responsibilities and skills should our listeners adapt? Or does this just mean that they don't have to develop engineering skills? Yeah, so how does it change things for our listeners who are maybe developing data applications, developing models for deployment? And then after you've kind of thought about how it changes things for an individual, how does it change things for an organization in terms of research engineering operations, something like Marmo seems to break down a lot of silos.

Akshay Agrawal: 18:23

Yeah, that's a really good question. I think it's an astute observation. So in terms of the individual, the practitioner, data scientist, or even ML engineer, AI engineer, data engineer, I think the way that I think about it is that Mamo gives them new capabilities to make their work just far more useful in the organization. And that gets into your second question too. How does that change things within an organization? So the data app is one good example. Previously you may have done your experiment in a notebook and then you're like, okay, where's the front end engineer who can help me actually build an application around this to make my work actionable? At best, you might try to reach for something like streamli, but you would hit performance bottlenecks there rather quickly. It's like with marmo, there's no migration phase like your notebook should. You want it to just change a couple of variables to sliders or dropdowns or tables or whatever you need.

19:22 And then all of a sudden you have a data app that you can then share with your team, but also to your CEO. So actually in a number of companies that are using US, CEOs are using Marmo notebooks that their data scientists made for them. Sometimes the CEOs are actually making their own marmo notebooks to run their own operations just because the barrier entry is so low and it's even lower once you consider LLM integrations that we have just five code your way through a pretty simple tool that you can make. So that is one way that I think mima gives the data scientists new capabilities and also brings new people sort of into the fold. There is another way, and it has to do with what you mentioned of Murma notebooks being stored as Python files. Because actually every Mariama notebook is an executable script.

20:18 It's just the Python file. You can go to the command line and say, Python, my notebook py, even pass the command line arguments. So from your notebook, now you actually have a workflow that you can run as a pipeline or as a Quran. And so that's yet another way. By reducing that friction, we've now made you hopefully a lot more productive. And also there's a lot of alsos, but because it's a Python file, you can actually say from my notebook, import my function or import my class. And when you do that, that means that there's famously, for many years now, people have talked about the notebook to production handoff, the researcher to engineer handoff. That makes that handoff a lot less difficult, a lot more streamlined because as Marmo nudges you to write better code and it gives you these reusable functions, you can actually give your notebook to someone and they can import it and just use the logic. You could even write tests for your marmo notebook. Marmo works with pie test. So

		that's another way that it makes notebooks actually more useful in the engineering context as well.
Jon Krohn:	21:30	Alright, that's it for today's In case you missed an episode, to be sure not to miss any of our exciting upcoming episodes. Subscribe to this podcast if you haven't already. But most importantly, I hope you'll just keep on listening. Until 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.