SDS PODCAST EPISODE 772: IN CASE YOU MISSED IT IN MARCH 2024

Jon Krohn:	00:00:05	This is Episode #772, our first-ever “In Case You Missed It” episode.
	00:00:27	Welcome back to the Super Data Science Podcast. I'm your host, Jon Krohn. For this episode, we're trying something new: an "In Case You Missed It" or ICYMI episode that highlights the best parts of conversations we had on the show in the last month. This is a popular episode type on some other leading podcasts because it allows you to quickly evaluate whether there's a recent episode you missed that you should go back and check out. Alternatively, if you are one of those dedicated listeners that never misses an episode, this will reinforce the most important recent conversations that we've had in your memory, which could be useful for your career or to you personally. All right, with no further ado, let's get into it.
Jon Krohn:	00:01:07	So, first off, here’s bestselling author Dr. Sebastian Raschka on how Lightning AI makes LLMs easy. Dr. Raschka was our guest in episode number 767.
Jon Krohn:	00:01:17	Beyond your book writing, which is obviously heavyweight, you're an absolutely world-class publisher, a writer of books in this space, so that in and of itself could be conceivably a full-time job, but you actually have a full-time job. You have a day job, which is at Lightning AI. So Lightning AI is the company behind PyTorch Lightning, and you have a staff research engineer role at Lightning AI. So maybe tell us a bit more about what Lightning AI does and what your work involves there as a staff research engineer.
Sebastian:	00:01:58	So at Lightning AI, I am currently a staff research engineer where I work mostly on open-source and research, the intersection between open-source research and LLMs. So about Lightning AI. We are a company, a startup company headquartered in New York City, and we

have also a lot of meetups there, so if you are ever interested or in the area. We have multiple projects. Let's start maybe at the open-source, on the open-source front. So PyTorch Lightning is essentially an open-source library on top of PyTorch, and it is this library that really makes PyTorch more convenient, I would say, where it is not really, I would say, changing your model.

00:02:41 So typically, what you do is you define a module that you put ... So you put a PyTorch model into this module, and then you can use a trainer. So the trainer, you may be familiar with the concept of a trainer from other packages, but that's like the original trainer in PyTorch Lightning where it provides you with a lot of free goodies. So once you use this trainer, you get something like checkpointing, logging, easy access to different loggers, also available ones like weights and biases, but then also with one line of code, you can change whether you want to use one or four GPUs, eight GPUs, multi-GPU training versus multi-node training, different precisions, mixed precision training, and so forth. So these little things where it is a lot of code if you would write this out in PyTorch, so it's like a lot of boilerplate code and PyTorch Lightning basically packages that for you so that you don't have to, let's say, repeat or redo all that long code. You have a way of organizing your code in a more concise way. So that is PyTorch Lightning.

00:03:47 So the other project is called Fabric. Fabric is essentially the engine that is used for multi-GPU training that plugs in, for example, DeepSpeed, FSTP from from PyTorch. It's essentially this project that is with a few lines of code, you can manipulate PyTorch code. I have a few tutorials I could maybe share where it's really just five lines of code you change, and then you can easily access more advanced multi-GPU paradigms and so forth.

00:04:15 This is especially useful when you also work with large models like LLMs, where you need to shard the models across multiple GPUs because they're too large to fit on a single GPU. If you want to, let's say, lower the precision from 32-bit to 16-bit or beef load 16 -bit or mixed precision with 30-bit and 16-bit and so forth or even quantization. So that is where you can do that with a few lines of code instead of just changing 20 or 30 lines in your code where it becomes a bit hard if you want to just do comparisons. So that is basically to make our life also easier as researchers when we work on things so we can focus more on, I would say, the model itself rather than the training loop and so forth.

00:05:01 The third project is Lit-GPT. So Lit-GPT is using PyTorch Lightning and Fabric, and this is an open-source library to basically have an implementation of LLMs that is somewhat human-readable. So it is essentially mostly scripts or script-based, but it is meant to be easy to maintain and easy to read. For example, there are lots of LLM libraries out there, but as the researcher, when I try to, let's say, use them, they work great as a user, but then if you want to change a few things about the architecture, it becomes a bit more tricky because there are so many files and so many dependencies and a lot of code, basically, that you would have to understand to make changes, and that was meant more for, I would say, as a toolbox for us internally to experiment with LLMs. It's also open-source if others want to use it.

00:05:57 It was, for example, one of the most widely used packages also in our repositories in the LLM efficiency challenge at NeurIPS in 2023, but I would say it's not reinventing the wheel in terms of LLMs because it's essentially about loading existing LLMs like Llama, Falcon, and so forth. So you can use other LLMs in there, but it is like this certain way of making it hopefully a bit more easy to experiment with. So when you do something like LoRA, QLoRA, fine-

tuning and so forth, that is all on the open-source side. That is usually what we use ourselves when we work on research projects, for example.

Jon Krohn: 00:06:38 Next up is Dr. Travis Oliphant, creator of the ubiquitous NumPy and SciPy libraries, on the future of scientific computing. Dr. Oliphant's episode was number 765.

00:06:50 Siri, Starkey, Lanam, you had some amazing questions, but Svetlana Hansen, we're going to get to yours. So, Svetlana, she's been a long time listener, many years now of participating in the podcast. So, she's a senior software engineer creating real-time network engineering solutions for NASA Spacecraft, which is a pretty cool job. And so, unsurprising to hear somebody working at the frontier of space asking a question like, "Where are we going with the future of scientific computing and Python libraries?"

Travis Oliphant: 00:07:25 Great question. I think I was capturing it when I talked about, essentially, the compiler framework. I see the future in something like torch.compile, in Numba, in JAX. Now, it's not there yet. I wish I could tell you, "Just do this." I would say though, write your code at a higher level. If you're writing low level loops, question why? You should be writing array code. I still think the future is array or array computing. Write your code with, "Here's an array, or a data frame, or database computing," because then it enables you to then, you're writing code in a way that can be optimized by compilers in the future. As opposed to the more detailed you write it, like, "Oh, I'm writing this loop and grabbing that element of this loop," and the more detailed you're doing that, the harder. Today you can do it. High level array computing, write a Numba of uthunk. Write a uthunk to do the low level code you can't just do with operations that exist. You do that and that will be future-proofed for the future. And it works today, and in the future, you can take advantage of new innovations and compilation technology. So, where

we're going, I think is a world where one, people write in a higher level and the code is faster still. It's going to take time to get there still, but I think there's ways to do it today. And two, I do think generative AI is going to help us with better human computer interfaces, so that more people will write even higher level.

00:08:49 You can just use English language to express your ideas, and then that'll translate to frameworks that will spell that out, that then get edited and, depending on where you are in the cycle of the innovation cycle, you'll either be operating at the very most exploratory part of writing human language and interacting with the visuals, or supporting the technical libraries that emerge from that exercise and making sure that they keep working, or even building frameworks at the lowest level. You have a whole career path, and you can kind of go up and down if you'd like, but I really see a world where I really want experts to be able to think about their problem. Python's been popular because it got out of the way of people, let them think about their problem instead of their pointer arithmetic and their semicolon placement. They could just think about their problem. And that's still needed. We have scientists solving hard problems, thinking at a big scale, and I want to support it, and I think you can. But keep track of the compile world, ask where this run is happening. Is there an optimizer somewhere that's able to take your code and then make it faster? And if the answer is, it's nowhere, then okay, maybe you should be thinking about ways to do that, or looking for cooperation plows to do that with.

00:10:03 And then otherwise, it's still frothy. Interoperability I think is still... And data. The other thing I would say is, don't lock your data up in things you don't know. Make sure you know or a public spec exists for your data. The future is going to be bad for you if your data is locked behind a proprietary format that is not available via lots

of people. So, that's the other, I guess, thing I would say. And you can use a proprietary database, you can use a compute engine that you pay money for, that's fine, but just make sure the data you rely on has some existence in a public forum.

Jon Krohn: 00:10:43 And next from episode number 763 is the award-winning, A.I.-focused venture capitalist Rudina Seseri letting us know what it takes to get a VC firm to invest in you.

00:10:54 So, Mohammad Raza who does audit data analytics at BDO Advantage. One of his questions is one that I was going to make sure you didn't get away from this podcast interview without answering, which is, how do you, what do you look for in potential investments in AI startups that you're looking at? So you must get hundreds of times more pitch decks thrown at you than you actually invest in. What is the difference between the companies that you invest in and that you don't?

Rudina Seseri: 00:11:26 So some things don't change. Others do. In the categories of some things don't change, the number one thing or facet that any venture capitalist, focus on AI or not, looks for is an incredibly strong founding team that doesn't just have a strong vision for the company they intend to build, but also unparalleled execution capabilities. The ideas are important. Execution is even more important. There is no wave of disruption, at least not to date, that has done away with that need. So I look for that every single time, and it manifests itself in different ways. If it's a first time founder, it's the hustle, is what else they have done in life and how they set XYZ mark and then they beat it. If it's a repeating founder, they have different characteristics and then other proof points, if you will. But I look for that execution ability or extra amazing execution ability in every case.

00:12:47 I look for founders that understand the problem that they're solving, very, very well. And understanding the problem well is an interesting notion. It could mean that they come from industry and from the industry that they're trying to solve the problem for, or that they are an outsider in. So they function as a disruptor to how the way things have been done in a particular space. But they also surround themselves with outsiders who are the adopters to get that perspective. Both work, both fail, but both work, but importance around understanding the space. And here's why. You could have an incredibly strong tool and platform in your view or in absolute terms, and I'm going to give you an example in a second, but it fail because it doesn't follow the workflow of the users or adopters, or it requires a mindset change or behavior change. Those are hard.

00:13:58 So one of our partners at Glasswing is Vlad Sejnoha, the former CTO of Nuance and sort of godfather really to natural language understanding and speech recognition. And he often points to the example of medical transcription where in the early days there were tools that could do away with a human transcribing the dictation notes of doctors and doing away with that by leveraging software that was 80-plus percent accurate. And you think, "Great, 80% accurate. Oh my goodness." If you talk to Vlad, he tell you that it was a total disaster and it had to be refined because it didn't take into account how efficient these transcribers had become and in the traditional way versus needing to understand, oh, what was this word that the software or the algorithm did not understand correctly, et cetera. And in fact, the app had to become something like 90 to 95% in that range, accurate, for it to outweigh performance. So that's why I mean it really matters because it's in what context?

00:15:16 In another context, 80% might be off the charts good and beyond good enough. In others it is not. So

understanding the context of the application for the problem, the industry, the vertical, whatever the function, whatever the case might be, it's incredibly important. But execution and then understanding how AI works, again, it's as much an art and as a science as you're building the architectures. And what data do you have access to? So I look for modes. I look for modes around explainability. I look for modes around traceability. I look for modes around AI nativeness, and then execution.

Jon Krohn: 00:15:58 And building on the A.I. startup theme, our fourth and final conversation in this In Case You Missed It episode is with Prof. Zack Lipton on his roadmap from AI startup to long-term commercial success. Zack's episode was number 769.

00:16:14 You were talking from the very beginning about dozens of subjects being in a medical study when there are tens of thousands or hundreds of thousands of patients with that condition. And so it's conceivable that a tool like Abridge could someday be in real time saying, "There are 100,000 people in the United States currently afflicted with this condition. And of those, 20,000 were given this prescription and there was this outcome." That kind of thing becomes possible when these data are being collected and when models are being run over them. I'll give you a chance to comment, but I know I'm getting out of control.

Zachary Lipton: 00:16:58 No, I think that broadly, sort of everything is possible. And I think that that's sort of the magic of this company. We like to talk about it as a T-shaped company where you could think of like the stem of the T would be like, "If that's all you had, that would be like one of these other companies in the space that are more like just sort of pure commercialization efforts." The tip of the spear, that's the thing that has shot off and become a real full-fledged go-to-market operation, a commercialized

technology, something that's out there. But the base of the T is the more foundational R&D. If all you had was the base, you'd be a research lab. And if all you had was to stem, you'd be a kind of pure commercialization effort of like, "Let's just kind of grab the tools and wrap them in a software package and try to hit the market hard."

00:18:14 And I think we see that there is a special connection between these because we nail one problem and really do right by our customers. We're able to reach a level of quality that other people wouldn't be able to have if they didn't have the same caliber of scientists working, the same kind of technical acumen in the company. But also, we're able to learn from that data, we're able to come back and potentially... Whether you think of it as a new product line or you think of it as expanding the offering, we're able to develop new technology. And some of that effort in the kind of foundational R&D level is pursuing ideas that might be transformative in 10 years, but some of it is pursuing ideas that could be relevant, could be reshaping our current offering on the scale of weeks to months. So I think that's...

00:19:25 I think the thing that's been special and that is really important to me to keep is that somehow we have this thing of, we have the foundational science happening, we have the commercialization happening, and we have all of them kind of informing each other. And we've been able to do it in a way where it's not like a siloed company. Because this is, I think, the failure mode that I've seen so many times at companies, is that you wind up in a place where research is this completely segregated out unit and the definition of a good manager in research is someone who keeps anyone associated with a product from coming anywhere near you. And it's like that kind of company, where it's like your main job is just maybe to not touch the company data, to not really think about their business problems, to publish some papers. The value

prop is you make the company look good, there's reputational benefit, they get to be like a leader in AI, but there's this weird gap that forms of friction.

00:20:30 Oftentimes, there's even a weird elitism or hierarchy of the product people think the research org folks are out of touch, the research people think the product people are not... whatever, whatever. And obviously, I think this challenge, it's easier to do it when you're a smaller company. So, I don't mean to, by any means, be like, "I figured it all out and Meta hasn't," or something. It's much harder on a larger scale. But to create a certain kind of energy where every single scientist feels directly dialed into the mission of the company, is directly aware of what's happening, and deriving insights and inspiration from what we're seeing in the field and is even [inaudible 00:29:58] wired.

00:21:16 And we've been very clever about how we design the processes around how feedback is collected and where it's channeled so that everyone's wired into it. It's also a lot easier to get excited about it when the form of feedback is you're getting notes from the field, from clinicians. It's written feedback. Often it's personal. It's insightful. Sometimes they're even designing features with us, because they're like, "Oh, I would've liked it if the app did this, this, this." And sometimes it's like, "Oh, why weren't we doing that yesterday?" And other ideas, you're looking at it and you're like, "Well, that's a little bit sci-fi, but maybe you should take it seriously." And other ideas are kind of wild. But I think that given that... is you can get excited about it in a way that I think is hard to get excited about, like improving click-through rates or something like that. And that feeling of impact.

00:22:17 It's a strong selection criteria. We have to select on talent and skill and grit and hustle. But we select really hard on... if someone's sort of like, "I could just as easily be

doing this work for a hedge fund," or something. It's sort of probably not a great fit for Abridge. And I think because we've selected on a certain kind of mission alignment, that's the glue that holds together the kind of more foundational research, the near-term research, the sort of translation of models into a live product, the app engineers, the customer success folks, the go-to-market team. It's strangely cohesive. Every other organization I've been at, you're like... there's this sort of like, "Engineers keep the business people away from me." And I think here we have this kind of energy that is made possible by a kind of belief in what we're doing.

Jon Krohn: 00:23:12 All right, that's it for today's episode, our first-ever In Case You Missed It episode. Since this is something new, don't hesitate to reach out to me via a LinkedIn post or Tweet to let me know what you think or comment on the YouTube video. Whether this new format worked at all, let us know, what we could do to improve it, and so on. Any feedback at all is always most welcome. Otherwise, be sure to subscribe if you haven't already and, most importantly just keep on listening. Until next time, keep on rockin' it out there and I'm looking forward to enjoying another round of the Super Data Science Podcast with you very soon.