SDS PODCAST EPISODE 693: YOLO-NAS: THE STATE OF THE ART IN MACHINE VISION, WITH HARPREET SAHOTA

Jon Krohn:	00:00:00	This is episode number 693 with Harpreet Sahota of Deci AI. Today's episode is brought to you by AWS Cloud Computing Services, by WithFeeling.AI, the company bringing humanity into AI, and by Modelbit, for deploying models in seconds.
	00:00:20	Welcome to the SuperDataScience podcast, the most listened-to podcast in the data science industry. Each week, we bring you inspiring people and ideas to help you build a successful career in data science. I'm your host, Jon Krohn. Thanks for joining me today. And now let's make the complex simple.
	00:00:51	Welcome back to the SuperDataScience podcast. Today I'm joined by Harpreet Sahota for a special episode examining the state-of-the-art in machine vision. It's hard to imagine a better guest than Harpreet to guide us on this journey. Harpreet leads the deep learning developer community at Deci AI, an Israeli startup that has raised over 55 million in venture capital, and that recently open-sourced the YOLO-NAS Deep Learning model architecture. This YOLO-NAS model offers the world's best performance on object detection, one of the most widely useful machine vision applications. Through his prolific data science content creation, including his Artists of Data Science podcast and his LinkedIn live streams, Harpreet has amassed a social-media following in excess of 70,000 followers. He previously worked as a lead data scientist and as a biostatistician. He holds a master's in mathematics and statistics from Illinois State. Today's episode will likely appeal most to technical practitioners like data scientists, but we did do our best to break down technical concepts so that anyone who'd like to understand the latest in machine vision can follow along. In the episode, Harpreet details what exactly object detection is, how object detection models are evaluated, how machine vision models have evolved to excel at object

detection with an emphasis on the modern deep learning approaches. How a neural architecture search algorithm enabled Deci to develop YOLO-NAS, an optimal object detection model architecture. The technical approaches that will enable large architectures like YOLO-NAS to be compute-efficient enough to run on edge devices and Harpreet's top-down approach to learning deep learning, including his particular recommended learning path. All right, you're ready for this eye-opening episode? Let's go.

00:02:41 Harpreet, my friend. Welcome back to the SuperDataScience podcast. It's been over two years since your first episode. That was way back, episode number 457 in the spring, northern hemisphere, spring of 2021, and that was when I met you. Since then, although we've never met in person, I would say we're friends.

Harpreet Sahota: 00:03:01 Yeah, dude. Absolutely. Dude. So, good to be back on the show. I can't believe it's been two years and yeah, like me and Jon are on, on text message and phone call kind of basis. I hit him up whenever we need some advice with, with you know, whether it's a career move or whatever, just to say happy birthday and all that. So, we've been in touch over the last two and a half years, but can't wait to meet in person someday.

Jon Krohn: 00:03:21 Yeah, yeah. Now that the pandemic is over, I'm sure we'll run into each other at a conference or somewhere in the world soon, and I'm looking forward to it. So, when you were last on the show, the episode was focused on landing a data science dream job because you were, I mean, you still are very involved in encouraging people's careers, but you were doing it explicitly as part of how you were splitting up your week back then. At that time, you were also full-blown focused on the Artists of Data Science podcast, which is a very cool concept for a podcast. It's designed for a data scientist audience, but it's talking to guests that aren't necessarily data

		scientists, they're philosophers and writers, and ... so you're, you're really trying to like cultivate creativity and possibility, artistry amongst data scientists. So, really cool format. And I know that with all the things you got going on, you're not releasing those episodes as much anymore, but you've got a new podcast, Deep Learning Daily. Is that a daily show? Is it really daily?
Harpreet Sahota:	00:04:31	So, so Deep Learning Daily, it's so it's the name of the community that's Deci's kind of sponsoring. And so this community, we've got a, I've done a number of virtual events for it and recorded like podcast episodes. Those will all be released in due time. I've got like a backlog of 30 something episodes. It's just a matter of time to, yeah to edit it and all that. And I'll actually, I'll be at CVPR next week recording interviews with researchers and stuff, and that'll all be, you know, released on that podcast. But yeah, Deep Learning Daily is like the Discord community, and then it's also the Substack which is the newsletter, and that's where the audio and video will be housed for that. And then also it'll be on podcast platforms as well. I like, I, like this podcast is like almost a 180 from the Artists of Data Science, because this is completely technical. And we're just getting into the weeds of deep learning with people, which I found extremely fascinating. Big departure from the Artists of Data Science, but it's a good direction.
Jon Krohn:	00:05:33	Well, very cool. I remember over two years ago when we did record your episode, you weren't, you were very interested in deep learning, but you hadn't formally studied it that much. I don't think you'd done anything in production at that time with deep learning. But now because of working with Deci, you are full on into deep learning because they're such big deep learning specialists. So, it's exciting to see all the posts you've been making, and I'm looking forward to digging into

technical expertise in this show. So, yeah, back then you were a data scientist at a company called Price Industries, and now a couple of job changes later you are Deep Learning Developer Relations Manager, if I'm getting that correctly, at Deci AI. And so Deci AI is a deep learning acceleration platform. Very cool company actually invested in by George Matthew, who was a guest on the show here in episode 679. And so yeah, fast-growing company and what I want to focus on in this episode is specifically the groundbreaking foundation model that Deci released called YOLO-NAS. And so YOLO-NAS is a computer vision algorithm. So, we're gonna start with like the basics a little bit, and then we're gonna quickly ramp up into more and more technical detail so that hopefully all of our listeners can follow along. But so start us off by just explaining, Harpreet, what machine vision is.

Harpreet Sahota: 00:07:07 Yeah. So, when you think of like just traditional image classification right, you'll have, let's say some image, you present that to your neural network, and then you'll obtain a class label and some probability kind of associated with that prediction, right? So, you got one image in one class label out. And that's, that's great when, you know, the thing that you're interested in is maybe characterizing the majority of that image space, or it's like the most prominent thing in that image. But object detection takes that one step further because it's not only telling you what is in the image, like the class label, but also where in the image that thing is, right. And it's telling that via a bounding box. So you get bounding box coordinates. So, through object detection, you'll input an image, you know, video is, is still just a series of images, but you, you input an image and you obtain multiple bounding boxes and class labels as the output. So just kind of, you know, at the core of this, at, you know, object, any, any object detection algorithm kind of follows a similar pattern, right? The input is the image that you want to detect objects in, and then you'll get an output of a list of bounding boxes, which are the XY coordinates for each object in the image, the class label, and then some probability score associated with what the network thinks that thing is.

Jon Krohn: 00:08:31 Nice. Very cool. So, the kind of the classic machine vision task. So, 10 years ago, machine vision researchers, the state-of-the-art was mostly focused on classifying the whole image, like you were just describing. So, famous architectures like AlexNet released in 2012 was working on the ImageNet large-scale visual recognition competition data set. And they, in that competition you're just trying to get the whole image, right? So, like the image would be of a cat or a dog or a plane or whatever, and you'd want the algorithm to be able to identify that. With object detection that you're describing it's this more specialized task where there could be lots of things going on. There could be a cat playing with a dog. And so you have to be able to first put a bounding box, like you said, which is just a rectangle of any dimensions that, you know, so it could end up being that in the image it's just a big cat face. And then the bounding box would just be like, basically like the whole image.

00:09:30 Or it could be that there's like a, just a cat in the corner and there's nothing else. Like, I don't know, the whole rest of the image is just like plain brown. There's nothing going on. So, then you'd have a little bounding box in the corner where the kitten is. Or you could have a bunch of things going on a bus and a horse and a cat. And so you have these rectangular bounding boxes all over the image and then you're tasked with correctly identifying what's in that bounding box. So, it's a much more complicated task because you're, you're breaking down the big image into a bunch of smaller images and then classifying each of those objects that you find that you detect in the bounding boxes.

00:10:07 There's also to maybe help people kind of understand how there are like different kinds of machine vision tasks. There's also image segmentation, which is instead of being bounding boxes, it's pixel by pixel. So, every single pixel in the image gets classified as a particular category. And so yeah, like different kinds of approaches. I think image segmentation, like maybe that's better when you have a relatively constrained set of classes where like maybe in like a self-driving car application you're like, okay, we need to be able to identify like pedestrians, cars, road. And so you have kind of this relatively finite set of classes that you'd want to be able to bucket the pixels into. But I think that object detection, and I'm not an expert in this stuff, you might know more than me, but with object detection, you could have I mean, with modern deep learning algorithms, it's probably almost limitless. Like you could have a crazy number of possible objects you might even be able to do... whoa, I just thought of this, right? Like, I don't spend much time thinking about this Harpreet, but you could even, you could use object detection to find where the bounding boxes are, and then you could use like a clip style algorithm to just, you could literally describe anything like, then you have an infinite number of classes. Like, so that's kind of a, so anyway, I digress.

Harpreet Sahota: 00:11:33 I think that's, that's along the lines of that new model, I think Meta put out Segment Anything. So, that, that, I think that's right along the lines of that, I haven't gone too deep in the paper, but it's essentially what you're describing a promptable segmentation type of model. But yeah, segmentation, that's, it's an interesting thing is, it, segmentation is this concept of things versus stuff, right. Stuff is just the amorphous stuff in the background and things are this right group of pixels that are together that belong to a particular thing. Yeah.

Jon Krohn: 00:12:43 Are you stuck between optimizing latency and lowering your inference costs as you build your generative AI applications? Find out why more ML developers are moving toward AWS Trainium and Inferentia to build and serve their Large Language Models. You can save up to 50% on training costs with AWS Trainium chips and up to 40% on inference costs with AWS Inferentia chips. Trainium and Inferentia will help you achieve higher performance, lower costs, and be more sustainable. Check out the links in the show notes to learn more. All right, now back to our show.

00:12:47 Yeah. And so, so with object detection, which we're gonna focus on in this episode, you're identifying the things in, in these bounding boxes in the, in the image. And so tell us a bit about, there's in recent years, YOLO style architectures, which YOLO is like when you're a college student, I guess going on spring break, it's You Only Live Once. But with these architectures, it's, You Only Look Once. So, there was this original paper a number of years ago on YOLO architectures, but I think you probably know enough about this that you can even go back before YOLO and tell us kind of how object detection has worked historically at a high level, and then what these YOLO architectures changed and why those are like the dominant architecture for object detection today?

Harpreet Sahota: 00:13:36 Yeah. Even before like deep learning, there was, you know, people were doing object detection, but they're using, you know, something like histogram of gradients plus maybe some linear support vector machines on top of that to, to do whatever they needed to do. But of course, with classic ML, like you have to hand-engineer all those features, which is, which is a pain. But then right around 2014 is, you know, the AlexNet moment happened in 2012, and obviously people started going crazy over neural nets and deep learning. But 2014 was like one of the earliest object detection the earliest deep learning based approaches to object detection. That was called RCNN. So, that's Region-based Convolutional Neural Network. So that one, like you would take in an input image, then you'd extract a bunch of you know, what they called region proposals, then, you know, then you'd use some CNN to do some type of feature extraction and then classify whatever the regions are, right?

Jon Krohn: 00:14:36 Yeah, CNN being Convolutional Neural Network, which was like, and we get actually, so just to, to like really quickly digress. In natural language processing, which is my expertise really, we have now gone completely from using Recurrent Neural Networks, RNNs or LSTMs - Long Short Term mMemory networks. We now are completely in this transformer world where we're like, wow, transformers are way better because they can take into account like all of the context in a given piece of language. There are trade-offs the people are working on where like the compute complexity, squares polynomially when you're when you're working with these transformer architectures. But is, are CNNs still used a lot in machine vision today, or are those also moving over towards transformers?

Harpreet Sahota: 00:15:27 I think they're still being used for sure, but transformers have crept into, into the computer vision world as well. So, there's a lot of new transformer-based kind of architectures for detection and classification tasks as well. So, yeah, it's definitely creeping into there. I haven't done too much research into that. Like I've only really been doing deep learning for, for about a year. And so this is on my long list of things I need to figure out. But-

Jon Krohn: 00:15:55 Well, it's just a quick digression because I've, yeah. So, you're confirming for me that transformers are being increasingly used in machine vision, but CNNs are also still being used. So, that is a bit different from natural language processing, where like, I don't really know anyone who's using RNNs anymore. But anyway, so CNN, so you were talking about RCNN in 2014 being the first big deep learning architecture for object detection, and then I started interrupting you.

Harpreet Sahota: 00:16:19 Yeah, yeah. And then, yeah, there's improvements on RCNN, there's fast RCNN, faster RCNN. And the thing with these type of methodologies is that you needed to do two passes through your neural network - to classify and then get the bounding box coordinates. But then YOLO came along and YOLO kind of just, you know, changed all of that. So instead of having to have two passes through the neural network, you only look once, you only have to look at the image once and do one pass through the neural network. And so they became, they took off in popularity because their impressive speed and accuracy. So I don't know if you want to quickly just talk about kind of like the anatomy of an object detection model, like you've got the backbone neck and the head?

Jon Krohn: 00:17:04 Yeah.

Harpreet Sahota: 00:17:04 Right. So, the backbone is, is where we would extract features from an input image. And, you know, typically that's done using a convolutional neural network. And then it's capturing kind of hierarchical features at different scales. So, lower level features like, you know, edges and textures are, are being extracted in, you know, the more shallow layers. And then as you get deeper, you get more higher level features like, you know, more kind of semantic type of information. And then the neck is the part that is connected to the backbone and head kind of does that actual classification and bounding box prediction for you as well. Yeah, so YOLO came around in ... the paper was published in 2015, but it was presented at the CVPR conference in 2016. And it just, you know, took the world by storm with how fast it was and it smashed the previous state of the art. So, mAP -M A P, Mean Average Precision that's the metric that we look at when we measure how good the object detection model is. If you want to talk about that, we could, I can give it, you know-

Jon Krohn: 00:18:15 Yeah, let's, let's talk about that. I just, just really quickly I'm gonna get you to elaborate a little bit on CVPR. So, you've mentioned that you said how with your Deep Learning Daily podcast, you're gonna be there at the time of recording in the future. And so probably by the time this episode's released, it'll be in the past, but you're going to the CVPR conference, which is, it's the premier computer vision conference, right?

Harpreet Sahota: 00:18:40 Yeah, yeah. Computer vision pattern recognition, but really it's like a, a deep learning type of conference and yeah, one, like, one of the well-known ones kind of up there with NeurIPS and what's the other one, ICML.

Jon Krohn: 00:18:57 ICML, yeah.

Harpreet Sahota: 00:18:58 Yeah, big, big conference.

Jon Krohn: 00:19:00 But yeah. And, and I guess this one skews a bit more towards machine vision applications than those other two necessarily would. Super cool. So, yeah, so yeah, so you were saying, yeah, these, these kinds of big breakthroughs like YOLO get published in the CPPR proceedings, and then you were about to tell us about mAP, which is a key metric for assessing the quality of a object detection models output. So, I guess like there's these two kinds of tradeoffs that you're trying to work through. Like, you, you want it to be fast, and so you're, you've been talking about how from RCNN to fast RCNN to faster RCNN, obviously they're going faster, but then

YOLO was a big step change because it didn't require these multiple passes. It was able to, in a single pass, be able to both identify where objects are in the images as well as classify those images, which is kind of a mind-blowing thing to me. Like you, you described it there through the backbone, neck and head anatomy, but like, because I haven't spent enough time with it myself, it's still something that like, I'm kind of mind blown by. But, so obviously speed is a key consideration, but then simultaneously another big consideration is accuracy of course. And so I guess mAP is like the key way to measure that accuracy.

Harpreet Sahota: 00:20:23 Yeah, yeah. And it's based on precision and recall, so familiar terms to classical, you know, data science folks. So, so mAP gives you kind of a balanced assessment. And then there's another assessment that you would call IoU that this Intersection Over Union that tells you how good your bounding boxes is. And then there's another thing in there, an object detection that we call non-max suppression that helps filter out, you know, redundant bounding boxes that you might get during training. But so just kind of breaking it down, right? So, we all know what precision is right? It's just the model's ability to make an accurate, positive prediction, right? And then recall is the number of actual positive cases that the model correctly identifies, right? So, precision, you can think of it as like a sharpshooter and it's hitting the target accurately. And, and, and recall is like a detective who is catching all the suspects in the crime. So there's always that trade-off between precision and recall.

Jon Krohn: 00:21:17 Nice, I love that analogy. That is, that is easily the best analogy I've ever heard for explaining precision and recall. And maybe now like I'll be able to remember it and I won't have to keep going to Wikipedia every time someone's like, I'm like, yeah, I know what that is. Everyone's always talking about that. I of course know what it is.

Harpreet Sahota: 00:21:35 Yeah, yeah. Dude, trust me, like I get it confused all the time. Don't ask me the formula, but I don't know what the formula is off the top of my head. But so what mean average precision does is, you know, we have the precision recall curve and mAP - mean average precision is calculating the area under the precision recall curve to give us like a balanced assessment for how good this particular model is. So what it does is it's incorporating the precision recall curve, like I've mentioned, and plots precision at against recall for different thresholds, right? So when you do it this way, you get kind of a more, more balanced assessment because you're considering the area right under the curve. And then you got to think about the multiple objects that are, you know, being detected in an image.

00:22:25 So mAP is able to handle multiple object categories by calculating each category's average precision separately, and then taking the average across all the categories, right. So, it's measuring average precision for the object. You do that for all the different objects, and then you average that and you get the meat average precision. Then there's another concept to measure how good the object detection model is Intersection Over Union. And this just measures the quality of the predicted bounding box by comparing kind of the intersection and union of areas.

Jon Krohn: 00:23:00 Got it. So, the mAP, the mean average precision is for figuring out once, like with within the boundary boxes, how accurate are the predictions? And the IoU is more about how accurate are the placements of the boundary boxes.

Harpreet Sahota: 00:23:13 Yeah, yeah, exactly.

Jon Krohn:	00:23:15	That makes perfect sense. Sweet. So, you've given us an introduction to YOLO. What has happened since, because I think the original YOLO architecture you said was like 2016. And so yeah, since then there's been a bunch of different versions. There's like YOLO version two and YOLO version three. And I don't know if those were like incremental changes, but ultimately it leads to what Deci's been up to and just this year made a big splash with the release of this YOLO-NAS architecture. So, like, take us on the journey to YOLO-NAS.
Harpreet Sahota:	00:23:49	Yeah. So, yeah, the first YOLO, YOLOv1, like the paper was published 2015. It was presented at CVPR 2016. So, it's, you know, it's been around for a while. And since then it's been 16 different YOLO models have been released. there's this really, really good paper on archive called A Comprehensive Review of YOLO that goes from YOLOv1 and Beyond by Juan Terven and Diana Cordova-Esparza. And they recently just updated it a few days ago to include YOLO-NAS on it as well. 35-page research paper. But I summarized their findings in a edition of the Deep Learning Daily newsletter. But yeah, like there's been a bunch of YOLOs and you know, what characterizes all of them is just speed and accuracy. So, you know, the first three YOLOs, YOLOv1, YOLOv2, YOLOv3, these were all created by somebody named Joseph Redmon and Ali Farhadi, I believe his name is. These are the original creators of YOLO. Redmon left computer vision research for, you know, ethical reasons after YOLOv3. But people have kind of just adopted that name as you know, as a framework, there's, there's this brand affinity with, with the YOLO name.
Jon Krohn:	00:25:12	Did you say, so I don't, I don't, I don't know if it was just like a, like a glitch in the recording or what, but just as you were saying why Joseph Redmond left, did you say for ethical reasons?

Harpreet Sahota:	00:25:24	Yeah, for ethical reasons. He was not he, he was not happy about the application of his research for military purposes. And you can obviously envision the military purposes that somebody would use object detection to target things and blow up.
Jon Krohn:	00:25:38	Yeah, yeah. Yeah, this stuff like, okay, like detect, you know, what kind of plane this is. Oh, it's it's a passenger airplane versus it's a Russian Mig and yeah and then it can be used to... yeah, I don't know the extent to which they are, like automated systems in terms of like actually firing it or if there's a human in the loop, but obviously yeah, there's some pretty concerning ethical implications.
Harpreet Sahota:	00:26:04	Yeah. Yeah. And so he, he loved computer vision research, and then somebody named Alexey Boch-kovsh, I can't say his last name, Bochkovskiy he started off with YOLOv4. And so YOLOv4 hit the ground and after YOLOv4, it was YOLOv5. So, so, okay, so YOLOv3 was originally, I think it was like in the Darknet framework for C++, but then an engineer named Glen Jocher took YOLOv3, ported it over to PyTorch, so it became available to the PyTorch community. So then Glen then created YOLOv5, you know, completely new architecture release it at in, in, you know, this PyTorch kind of model. There's a bunch of controversy about that, don't want to delve into that too much.
	00:26:57	But yeah, since then there's been a number of YOLOs, like ScaledYOLO, YoloR - You Only Learn One Representation YOLOX, which is exceeding YOLO series in 2021. there's the YOLOs that came out of the PaddlePaddle research group in China. Then, you know, Glen and Ultralytics published YOLOv8 earlier in 2023, January, 2023. But really like, you know, the prior to YOLO-NAS, the real state-of-the-arts was YOLOv6, YOLOv7, and YOLOv8, right? So, our model was inspired by YOLOv6 and YOLOv8, some of the blocks that they

		had in there. We kind of fed that to our neural architecture search algorithm and ended up with YOLO-NAS. So, what is YOLO-NAS? YOLO-NAS is in a nutshell, it's an object detection model. A new state-of-the-art.
	00:27:50	And it's outperforming YOLOv6 and YOLOv8 in terms of meaning average precision and inference latency. So that means it's more accurate and it's faster, and it's improving upon some of the limitations of the previous YOLOs. You know, previous YOLOs didn't really have adequate quantization support. You know, the tradeoff between accuracy and latency wasn't the best. And, you know, we're able to now be faster in real-time detection as well. Not only that, YOLO-NAS is, you know, supports intake quantization. So it's just the natural next step and it's, you know, it's not gonna remain state-of-the-art forever. Like, object detection is a super competitive like, field of research. There's people around the world working on this. You know, I'm sure somebody will beat us soon enough, but, you know, we'll be, we'll be right back at it.
Jon Krohn:	00:28:46	That's why it was so urgent that I get you on the show now. I need this episode to be live while you guys are still the number one object detection algorithm.
Harpreet Sahota:	00:28:54	Yes, exactly.
Jon Krohn:	00:29:38	The future of AI shouldn’t be just about productivity. An AI agent with a capacity to grow alongside you long-term could become a companion that supports your emotional well-being. Paradot, an AI companion app developed by WithFeeling AI, reimagines the way humans interact with AI today. Using their proprietary Large Language Models, Paradot A.I. Agents store your likes and dislikes in a long-term memory system, enabling them to recall important details about you and incorporate those details into dialog with you without LLMs’ typical context-window limitations. Explore what the future of human-A.I.

		Interactions could be like this very day by downloading the Paradot app via the Apple App Store or Google Play, or by visiting paradot.ai on the web.
	00:29:41	So, yeah, so super cool. So, YOLO-NAS is the fastest, most accurate object detection algorithm yet. Awesome. That you're saying that that means it's so fast now that it can be used in kind of real-time applications. And so, I think a key thing here is we know YOLO stands for You Only Look Once, but what does the NAS - N A S stand for in YOLO-NAS?
Harpreet Sahota:	00:30:02	Yeah, so that stands for Neural Architecture Search. Because the way this, this architecture was discovered was through this you know, Auto ML kind of technology called neural architecture search. Typically, you know, people discover architectures, they're doing tons of research and all that. Well, we just looked at what was out there, what worked, input that into our giant AutoNAC engine and got this architecture. So let's just kind of talk a little bit more about neural architecture search. So, what is this thing trying to do, right? It's, it's trying to find like an optimal network architecture for a specific task. Like, for example, that task could be detection, classification, segmentation, whatever. And what, you know, neuro architecture search does, is that it's automatically searching through possible architectures. So in the case of YOLO-NAS like our architecture we search space, well, we'll talk about search space in a second here, but our architecture's search space was 10 to the 14 different architectures.
Jon Krohn:	00:31:06	Whoa
Harpreet Sahota:	00:31:06	Ton of architectures, yeah. So, instead of like, you know, relying on manual trial and error or human intuition, you know, NAS is, is using optimization algorithms to find the architecture so that we're balancing accuracy, FLOPS you

know, floating point operations of the computational complexity, and like the actual size of the model. So, you know, how is it doing this? Well, you know, the search algorithm could be as simple as grid search or random search, or it could be more complex like Bayesian genetic reinforcement learning. But let's kinda just talk, talk about neuro architecture search though. So, we need, we need basically three things to make this happen. A search space, search strategy, and then some way to estimate the performance of the architecture that we end up with. So, the search space itself - this defines all the possible sets of architectures that the, that our algorithm can explore.

00:32:01 And so what's the search space consist of? It could be as simple as the number of layers in a network, or it could be as complex as the types of layers, the types of blocks, the connections between layers, various other hyper-parameters. You know, all the different, you imagine like, you know, you're building a Lego house, right? Like there's a myriad of different Lego pieces that we have, right? If you are trying to maximize the square footage of your Lego house by using the optimal blocks, right? This is what neural architecture is kind of doing at a high level. Like intuitively we're trying to find the right blocks to maximize something. So at Deci, like the thing that we have the AutoMac it takes it just a step further because in addition to everything I mentioned before, we also consider the hardware that you're deploying on and your data characteristics.

00:32:54 So the hardware could be, you know, in the case of YOLO-NAS we optimized it for the T4 GPU, which is industry standard for detection. You can even look at, you know, compilers, quantization as well. But you know, the, this search space is, it influences how the end architecture, you know, ends up being. So then we got a search space, but then now we need a way to search through this space because that's a ton of different pieces. And so again, you know, various methods, random search, Bayesian search reinforcement learning, evolutionary algorithms, gradient methods, whatever and this impacts how long you're searching for. Once you got those in place, then you need to have like some way to estimate the performance of your, of your out outcome architecture. And so this could be as simple as just training each architecture that you end up with on the dataset that you're intending to use it for, and just measuring the performance. Or you can do more advanced techniques that I really don't know how these work, but like curve extrapolation, One-shot NAS, weight sharing, things like that. But you put all of that in, right? That's what goes into NAS, the search space, the search strategy, performance estimation strategy, and then the output is an architecture that's optimal or near-optimal according to whatever metric you have in a nutshell.

Jon Krohn: 00:34:24 Very cool. That was a great explanation. I love the Lego analogy. You are the king of analogies. To make this, to take, yeah, difficult-to-visualize concepts and make them suddenly, instantly visualizable. So, very cool. So, all of this neural architecture search, all of this NAS concept, this is something that Deci has developed, right?

Harpreet Sahota: 00:34:48 Yeah. So, yeah, neural architecture search, it's, it's an active field of research. the thing that differentiates Deci's neural architecture search is the actual algorithm itself. So that's what's proprietary for, for Deci is our algorithm-

Jon Krohn: 00:35:01 Got it. Got it, got it-

Harpreet Sahota: 00:35:02 Neuro Architecture search. Yeah.

Jon Krohn:	00:35:05	So, NAS is both, like, the name of a field of research as well as like in capital letters, a specific algorithm that Deci's developed.
Harpreet Sahota:	00:35:14	Yeah. In our case, we call it AutoNAC, Auto N A C, Neuro Architecture Construction. That way we can put the TM on there.
Jon Krohn:	00:35:24	Got it. Nice. So, AutoNAC is Deci's proprietary NAS algorithm. Perfect. That makes a lot of sense. So, that was an amazing tour Harpreet of computer vision, object detection, and then the architectures that have led us to the state-of-the-art YOLO-NAS architecture today, including the AutoNAC approach that allowed the neural architecture search to identify this optimal architecture. And so if there are people out there who would like to learn more from you about computer vision, I understand that you are working on an intro to computer vision course, which is expected to be out in September or October of this year on LinkedIn Learning. That's cool.
Harpreet Sahota:	00:36:11	Yeah, yeah. Doing it on LinkedIn Learning, it's, it's, it's gonna be a cool course. So, like it, the audience for this course are people who are like me before I got into deep learning. So, if you're comfortable with statistics, math, Python programming, classical ML, if like, you're good with all that, and you're like looking at this deep learning thing and wondering like, okay, how, how can I get into this? Then this is the course that I made for you. I made it for an earlier version of me. And it goes through, like I start with like a history of computer vision for image classification, and I talk about, you know, important concepts like the things that I felt I needed to understand before I got into deep learning. So, I kind of structured it that way. I start from pre-deep learning methods, just briefly touch on those.

00:36:53 I talk about the importance of ImageNet because I didn't know what the hell ImageNet was, like when I first got into deep learning and the importance it had and why it's important. And then go into like AlexNet, and then I talk about a few different architectures there, like kind of fundamental architectures you know, I pay homage to AlexNet and LeNet. But also, you know, we talk about ResNet, EfficientNet, MobileNet, and RegNet as well. I think I talk about that. And we do everything. It, it, you know, it's, I do it devoid of much math. I try to make it as math unintensive as possible. And the example projects are all done using the Super-Gradients training library.

Jon Krohn: 00:37:41 Nice. I've never heard of that before. Super-Gradients training library?

Harpreet Sahota: 00:37:46 Yeah. Yeah. It's it's Deci's library. It's the official home of YOLO-NAS. And it's a, it's, it's PyTorch-based training library. It's just, it includes a bunch of like training tricks right out of the box, and it just abstracts a lot of the workflow, a lot of the boilerplate, so you don't have to write it. So, I mean, I know you're a huge fan of PyTorch Lightning. You can think of it kind of like that.

Jon Krohn: 00:38:07 Cool, yeah. So, it's like a wrapper around PyTorch that lets you do things more quickly when you're needing to be training models. So, like in base PyTorch, for example, like, you need to even be writing the structure of your loop through epochs of training and then each step within the epoch, but PyTorch Lightning abstracts that way. So, this does a similar kind of thing and is particularly for computer vision or it's, it's quite general,

Harpreet Sahota: 00:38:33 Yeah. Theoretically, you can use any PyTorch model, like any ML module for Super-Gradients, it'll work just fine. But our model ZOO, the pre-trained models that we have, they're all mostly computer vision at the moment. And we got pre-trained models for classification, detection, segmentation, and Pose estimation as well. And then we'll be expanding further in the near future.

Jon Krohn: 00:38:59 Nice. Very cool. Yeah, you can see how that's kind of CV focused with stuff like Pose estimation, which I'm assuming is like predicting what kind of pose a person has in an image or video, which is obviously gonna be specific to CV. Very cool. So, a concept that you've mentioned a couple of times and that I've touched on in recent episodes of the show as well, but I'd love to hear you tell us more about in the specific context of YOLO-NAS, is this idea of quantization. So, quantization allows us to reduce the size of our model by, instead of using full precision data types, we use half precision or maybe even less precision. So, you're, you're reducing the amount of memory and compute required for say, the parameters in your model. And so yeah, fill us in on the YOLO-NAS approach. My understanding from my vague understanding of YOLO-NAS is that it uses something called a Hybrid Quantization method. So, maybe fill us in on what that means relative to standard quantization.

Harpreet Sahota: 00:40:14 Yeah, yeah. So, like, exactly like you said, like quantization is, we're taking the weights in our model from like some high precision floating point representation, like 32bit, for example, to some lower precision representation, 16, 8, 4bit, whatever. 2bit, I'd be really impressed. But it, you know, reduces the model size and increases inference speed, but you suffer inaccuracy, like you pay for it somewhere. But this is a great approach because you're gonna be able to deploy that model on hardware with limited, you know, computational resources. So that's kinda like the standard naive quantization method. Then there's the hybrid quantization method, which is a little bit more advanced form of quantization. And in this case, you're selectively applying quantization to different parts of the architecture based on the impact of the model's performance on the hardware. So, it's, it's more of a selective approach that helps maintain the performance of your model, but you still get the benefits you know, reduced model size and increased inference speed.

00:41:19 So, within the context of YOLO-NAS, this hybrid quantization method, it's like selectively quantizing specific layers in the model. And so it's trying to optimize the accuracy latency tradeoff, while still maintaining performance as opposed to like, you know, standard quantization, which is gonna uniformly quantize like all the layers. And that causes more accuracy loss and there's pros and cons to both approaches, right? So, standard quantization, of course, we mentioned it reduces model size, increases inference speed, but drops in accuracy, right? So, it treats every part of the model the same, which is probably not always ideal. Hybrid quantization we're maintaining the model performance, but still reducing the model size, increasing the inference speed. But the thing is like it's more complex to implement and it involves identifying which parts of the model should be quantized and to what degree they should be quantized.

Jon Krohn: 00:42:20 Very cool. That was a crystal clear explanation, and you even kind of did the summarization back to the audience that I usually do, so I don't even have anything else really to add. Yeah, it makes perfect sense. So, yeah, quantization is reducing the complexity of the data type, and that does have this trade-off of accuracy going down when you're having the speed increase. And so, yeah, hybrid quantization is this cool way of selectively quantizing only parts of the model so that you get the speed increases without the accuracy loss. Super cool. And so I guess that kind of quantization could come in handy along with potentially choosing your model size.

So, I know that there are a few different YOLO-NAS versions. There's a, a small and medium and a large version. And so yeah, there's particular circumstances where you might want to be using one or another. I'm guessing that something like large is gonna be slightly more accurate, but a little bit slower, that kinda thing.

Harpreet Sahota: 00:43:22 Yeah. Yeah. The only difference between small, medium, large is just the number of parameters that that architecture has. So, a small version has about 19 million parameters. The medium version has like 51 million, and the large version has like, like 67 million, something like that. So those are just, you know, that's, that's all the small, medium, and large means.

Jon Krohn: 00:43:44 Deploying machine learning models into production doesn't need to require hours of engineering effort or complex home-grown solutions. In fact, data scientists may now not need engineering help at all! With Modelbit, you deploy ML models into production with one line of code. Simply call modelbit.deploy() in your notebook and Modelbit will deploy your model, with all its dependencies, to production in as little as 10 seconds. Models can then be called as a REST endpoint in your product, or from your warehouse as a SQL function. Very cool. Try it for free today at modelbit.com, that's M-O-D-E-L-B-I-T.com

00:44:23 Nice. And so there might be listeners out there who are thinking of particular object detection use cases that could be useful. Like maybe they work for a municipal government somewhere and they're like, oh, we could be using this to be monitoring traffic. Or maybe they work for a national park and they're like, we could be using object detection to be monitoring wildlife. And so they might want to be able to deploy their model onto a very small low energy device, maybe something that can be like solar powered. So, they might want to have their object detection model on like a Raspberry Pi or an Nvidia Jetson, these very, very small processors. Does YOLO-NAS like support that kind of, or like, is it particularly paired with quantization? Does YOLO-NAS get small enough for those kinds of edge devices? Yeah, fill us in.

Harpreet Sahota: 00:45:18 Yeah. Yeah, there's we, we offer 8bit quantized version of YOLO-NAS. So, you know, the full YOLO-NAS itself was optimized for Nvidia T4 GPU. We're working on making a version for the Jetson device. So we're almost there. It's a little bit more research work to go into it. But you know, we offer the 8bit quantization model that that could be used. But in general, like you know, like we can talk about the things to consider when, when we're deploying on these edge devices, right? Because like you mentioned, they're, they're small devices, right? So, they typically, they're not a full-blown laptop, for example, right? You can only fit you know, certain hardware on there and certain memory on there.

00:46:05 So, the things you have to consider are model size, right? Because the model that you have needs to be small enough to fit on the memory of the edge device and deep learning models, they get huge, right? They can get into the gigabytes. So in order for you to get that model to fit, right, you, you could be in the lab building the most accurate model ever, and it's amazing, but then you go to deploy, you're like, oh, shit, dude, this thing does not fit on, on the, on the actual device. So then you can look at using techniques like model pruning where you're just, you know getting rid of layers in, in the network, quantization, knowledge distillation. We could talk about knowledge distillation a little bit if you'd like. But you know, these techniques help reduce the size of the model without too much of a loss in performance.

00:46:54 So like I mentioned YOLO-NAS, it's int8, you know, quantazible to int8 which means I can reduce the model

		size to get on those edge devices. But that's not the only challenge you have. You also have inference speed, right? Because the model needs to be able to run quickly enough to process incoming data in real-time or near real-time, right? And so this, you can achieve by using, you know, efficient architectures and some of the automatization techniques that we talked about. In the case of YOLO-NAS, like we use AutoNAC to find the optimal architecture that was hardware aware for the T4, and it's considering all the components in the inference stack you know, compilers and then quantization and all. Then the other thing you got to worry about is power efficiency. Because like you mentioned, you might not have this thing plugged in, it might be running just hanging somewhere. It might be solar-powered, whatever. So there's a trade-off between accuracy and power consumption. Why? Because high model accuracy usually means more complex computations, more complex computations means more power needed to do those computations. And then finally it's just software compatibility, because you have to deploy this thing on that device. And there's, you know, what run time are you getting deploy on? So YOLO-NAS, it's compatible with like, like Nvidia TensorRT, which is a common one. Yeah, so, those are the considerations to make in those scenarios.
Jon Krohn:	00:48:27	Nice. Very cool. That was a really nice in-depth breakdown of the kinds of considerations we'd want to make as we deploy to these smaller devices. Very cool. I didn't know before I asked you the question that you had this level of expertise in it, so that's awesome.
Harpreet Sahota:	00:48:42	Definitely not, not an expert, just learning and listening and, and taking notes.
Jon Krohn:	00:48:49	Nice. Yeah. Clearly very well. You've, it's been awesome the level of depth that you've gotten into in all the topics

so far. So we've obviously talked about YOLO-NAS a lot in this episode. People might be excited to use it. Are there like commercial restrictions for people using it, either you know, out of the box or fine-tuned to some particular object detection task that might be relevant to their users?

Harpreet Sahota: 00:49:17 Yeah, so the, so it, so bit of a non-standard license for YOLO-NAS, right? So, if you want to take our pre-trained model with its weights, you are free to use them. But if you're using it for commercial purposes, then you need to get permission from Deci. So the pre-train model weights themselves are not open for commercial use. However, the architecture itself, the architecture itself is. Like you could take the YOLO-NAS architecture start from scratch and train it on your own data, right? And so that has like kind of its own that, that affects a lot of things is pros and cons with that as well. But yeah, long story short the weights from YOLO-NAS as they are are not available for commercial use. But you could take the untrained model with this architecture, trained it on your own data, and you're off.

Jon Krohn: 00:50:14 Nice. That's cool. That's a nice compromise. And we see that kind of thing with very large models that are released today. So, Meta's Llama architecture, for example for natural language processing, they were releasing it for people to use for academic use, but someone that they gave permission to do that with released the model weights to be available to torrent. And then but it's, but it, but you know, you still, if you are a responsible business owner, and you should be, if you don't want to get into a legal quagmire in the future, you then can't use Llama for commercial purposes. You'd be insane to. And so this kind of approach that you're suggesting with the YOLO-NAS where people can take the general architecture and then put the considerable expense in that you guys have and deserve some reward for, if they want to put that considerable expense in to train the model themself, fine-tune it to their particular use case, go for it. I think that that's a really nice compromise.

Harpreet Sahota:	00:51:21	Yeah. Yeah. And I mean, it's, it's not always easy to train like models like that right out the bat. So, you know, sometimes it makes sense to kind of just fork over whatever it is and get the pre-train weights.
Jon Krohn:	00:51:35	Yeah. And yeah, they really get to like, there's, there would've been a huge amount of compute required in the AutoNAC neural architecture search that you guys did. And so now that all of that huge amount of compute has already been done and this optimal object detection and architecture exists, you've already left people in a really great place. Yeah. Even if they want to just be, yeah, going from there and, and coming up with a commercial use case. Very cool. Nice. So, a term that I hear a lot that I talk about a lot in the context of these very large models whether it's like the Llama architecture we were just talking about which is specific to natural language processing, or whether it's a big machine vision model like YOLO-NAS is, we can talk about these as foundation models. So, Harpreet, can you fill us in on what that means to you and what makes YOLO-NAS a foundation model?
Harpreet Sahota:	00:52:32	Yeah, so foundation model to me, I kind of just go by the definition from that paper, and it's any model that was trained on broad data using semi-supervised or self-supervised techniques. And YOLO-NAS fits that bill. So, it was pre-trained on the Object365 dataset, and this has, you know, 2 million images and 365 categories and some crazy number of bounding boxes in that dataset. So this gives just a large number of images and categories, gives you a wide range of examples for this architecture to learn from. And this improves its ability to, you know,

predict on downstream kind of tasks. But we didn't stop with Object365 for pre-training. We actually went another pre-training round after that on pseudo-labeled COCO images. So, we took, so, okay, so pseudo-labeling, what does that mean? So, that's a semi-supervised learning technique.

00:53:35 So, what's semi-supervised learning, that's when you use a small amount of label data and lots of unlabeled data, pretty much, right? So, we had a pre-trained model, a model that we trained already on COCO, right? And it was our version of YOLOX, which we have available in the models for super gradients. We took our version of YOLOX and then on the COCO test set, which nobody knows the labels to, we labeled it using YOLOX, our version of YOLOX, and generated labels for that dataset. So, that gave us more data to train on - 123,000 more images to train on. So doing this, like, you know, we're able to use that test set to generate labels [inaudible 00:54:21] training. So this improves the models' predictions and ability to work on new data because now we're giving it even more data to learn from.

00:54:29 So, because it was trained on this, you know, such an extensive and diverse sets of data it's why YOLO-NAS has its high performance and generalization kind of abilities. Yeah, so I guess, you know, the intuition I guess behind that is that these architectures that you train on large enough data sets end up serving like as kinda a generic model that generalizes well for different types of downstream tasks. So we tested how well it works on the Roboflow 100 dataset which is this new benchmark made up of it's made up of a hundred different data sets from the Roboflow universe. And we performed well on that. We beat pretty much all the modern YOLOs. But then taking it a step further for the training, like we use some more training techniques like knowledge distillation, and then something called distribution focal loss. So I promised you guys one quick [inaudible 00:55:32] on knowledge distillation. So, just real quick-

Jon Krohn:	00:55:35	Yeah, yeah, you offered it and I had the opportunity, I was planning on circling back. Let's do it right now.
Harpreet Sahota:	00:55:41	Yeah. It's just, it's the process where you take, where, where you take, you, you know, it's, you have a smaller, simpler model that you call like the student. And when you train this student to reproduce the behavior of a larger, more complex model, which you can call the teacher. And so the idea is to transfer knowledge from the larger model to the smaller one. So, this way the student achieves higher performance than it would learning on its own. So, this helps create models that are faster and more efficient, but still have that high level of accuracy. And then distribution focal loss, if you want to talk about that, if your audience is interested, I could talk a little bit about that, but-
Jon Krohn:	00:56:21	Yeah, go for it.
Harpreet Sahota:	00:56:23	Yeah. So yeah. Okay. So, let's talk about that distribution focal loss. So, focal loss, what does that do? It modifies the standard cross entropy loss that we use for classification, and it's specifically designed to address class imbalance problems in dataset. So, the focal loss is a function that deals with class imbalance, where there's a lot of easy negative examples, for example, the background, and then there's relatively few hard positive examples, which are the examples that you want to detect. So, during training, a model might come across a large number of negative examples, and again, negative examples it's just parts of an image where there's really no object of interest you know, compared to parts of an image that have an object of interest. So this leads to imbalance, because model is now being overwhelmed by

easy negative examples, and it ends up not paying enough attention to these hard positive examples. So, the focal part of this focal loss function gives a higher loss for hard misclassified examples and the lower loss for correct, easy examples. So, it's focusing on the hard misclassified examples. So this way you're, it prevents a number of easy negatives from overwhelming the detector during training. So then just the distribution part of that distribution focal loss. Instead of calculating loss based on individual classes, it calculates loss based on the distribution of classes which improves the model's ability to handle class imbalance.

Jon Krohn: 00:58:05 Very cool. The focus there on the negative misclassified cases or you know, getting these misclassified cases correct. It reminds me of the way that gradient boosting works. So, for example, the way that we can take a forest of decision trees but each step of the way we're, with gradient boosting, we identify situations where the model was wrong and correct specifically those instances. And that's what allows XGBoost to be such a top-performing model approach, particularly for tabular data. And if people want to hear a lot about that episode 681 with Matt Harrison, we focused on XGBoost there. But yeah, it's interesting like this, that conceptual idea obviously is being implemented in the way that you just described in a completely different way, but that conceptual idea of taking where a model is wrong and specifically focusing on those instances to shore up the model in those cases, it makes a lot of sense.

00:59:04 Awesome. So, foundation models, knowledge distillation, distribution focal loss, you fill this in on a lot of technical concepts in the last few minutes, Harpreett, and, you know, that exemplifies in a nutshell, this enormous amount of knowledge about deep learning models and model training, particularly related to computer vision that you've developed in a relatively short period of time that you've been at Deci. So, clearly, you're studying super hard. And so it's interesting, this role that you have as a Developer Relations Manager. Some people might kind of think of that in their heads as being like, almost like a marketing role. And it is like, it is, like, the purpose of this is to help develop the community, but clearly there is a deep technical aspect to what you're doing. Like you are able to come on this show and be able to go into any, seemingly any level of depth on these complex deep learning topics. And so, yeah, I just, I find that interesting. And so I guess relatively quickly, what kind of data scientist would you recommend transition from a data scientist to this kind of deeply technical developer relations role like you have?

Harpreet Sahota: 01:00:28 Yeah. Developer relations, it's, it's like an umbrella term, much like data science is an umbrella term, right? Because when I think of data science, I think of, well, there's data engineering, there's, you know, business intelligence, there's data analysts, there's traditional data scientists, so on and so forth, right? Machine learning engineer, deep learning engineer, so on and so forth. Developer relations is the same way. It's an umbrella term for a set of kind of skills. And, you know, obviously community building is one of them. Advocating for the needs of your users on the product roadmap - the advocate is another. The developer evangelist type of role where you're one-to-many communications, right? Then, you know, the role I love is the developer educator type of role.

01:01:10 So the type of data scientist that I think would fit well in this role is, well you have to be comfortable doing, doing what I'm doing right now, right? Like, just coming up on, on stage, on screen multiple times a week and just being there and just, you know, being okay and comfortable with your own lack of knowledge because you have to learn a ton, right? Your community is interested in a number of different topics. You have to help them by creating content on those topics or bringing in experts on those topics. But when you bring in those experts, you got to get yourself knowledgeable so that you can have a good conversation with them, right? So it, it's really a, you know, you have to be, you have to be the type of person that just loves being uncomfortable, loves feeling like they don't know anything, and be driven by that. But then also just, you know, curious. You got to be curious about what's happening in the industry, the tools that are out there, keeping up on trends, all that stuff. So yeah, I'm having a hard time articulating the exact type of data scientist, but look at me. If, if you hear me, if you hear me talking and you feel like you know you like me, then you might be a good fit for developer relations. Yeah.

Jon Krohn: 01:02:27 Yeah, I think this ability to communicate clearly both orally as well as written is gonna be essential. And yeah, being comfortable continuously learning new topics. So, for a really broad range, because your community's gonna be interested in, in a lot of different topics. That makes a lot of sense to me. So at Deci you have specifically specialized in deep learning, and you've been doing that as this generative AI cornucopia possibilities has been taking off. So, what's that been like and what kinds of trends do you think will drive the future growth for artificial intelligence?

Harpreet Sahota: 01:03:10 It's been exciting being, you know, kind of in the space as this generative wave has been taken over. Like at, you know, we're working on generative use cases at Deci, like when we're at CVPR, we'll be showcasing our version of stable diffusion, which is it runs faster than normal stable diffusion. And I'm looking forward to us open-sourcing those models so that I could talk about them and create content about them and learn more about them. So it's just been a whirlwind like, you know, my Twitter feed is like the best Twitter feed ever. It's just nothing but like cutting-edge research happening. And, you know, you look at my, my read-a-later list and it's just deep, but I love it. I love it. And I guess in terms of trends, man I wish, I wish I could, I wish I was insightful enough to, to see what kind of trends that would drive the future of AI.

01:04:01 But I think, you know, deploying on resource-constrained devices is gonna be a thing, right? Like, yeah, we all have our phones attached to us, but you know, do we always want to send our data to OpenAI, right? I'm curious to see what Apple's gonna come up with that's gonna run locally right here on my phone that's gonna allow me to take advantage of these, you know, generative models on a small device like this. So, kind of the interplay between Internet of Things and generative models. That's what really, really excites me. Yeah.

Jon Krohn: 01:04:39 Nice. Yeah, that makes a lot of sense. And I've been blown away by the incredible capability of relatively small open-source large language models. So, for me, in the national language processing space, as I've talked about in a lot of recent episodes, and as I am working on daily at my company Nebula, we're taking open-source architectures, and these could be very small relative to the kinds of OpenAI LLMs that you're describing out there. So, the OpenAI LLMs like GPT-3 we know was 175 billion model parameters, and GPT-4 was probably larger. They never released that, but it takes longer to get GPT-4 results, so presumably, it's an even bigger model. So, these kind of state-of-the-art private models are hundreds of billions of parameters. But if you have a relatively constrained set of use cases that you need your model to be able to handle, you know, you don't need it to work in every imaginable language, for example.

01:05:43 You can take these open-source models that are often like 3 billion, 7 billion, 13 billion parameters. So, they're like from a 100th to a 10th of the size of OpenAI's private models, and you're able to fine-tune it very efficiently using parameter-efficient fine-tuning approaches, like Low-Rank Adaptation. And then you have this model that can be deployed on a single GPU. But as you point out, they've got to get even smaller, like being, like having a big cloud GPU as like, "hey, that's, it can even be that small?" that isn't nearly small enough. Like we need to be able to have these models run on people's phones or being able to run an on Raspberry Pi's or the Nvidia Jetsons' that we were talking about earlier. And so I think you're absolutely right. I think that the future of AI is smaller models that's approach the kind of performance that we see from models like GPT-4, but are constrained to more refined use cases. That makes a lot of sense to me. So obviously you're learning a lot about deep learning in particular, and I know that you have a particular philosophy of learning deep learning that you describe as top-down. Do you want to describe for our listeners what that means and why it might also be the way that they should be learning complex concepts like deep learning?

Harpreet Sahota: 01:07:08 Yeah, yeah. So, I'll, I'll preface this by saying that like, you know, I've got a master's in mathematics and statistics. I was a biostatistician, I was the actuary, I was a data scientist. So, there is, this is coming from that perspective, but even with, as somebody who has that background, like my approach is to just skip the math. First, skip the math, right? Ignore it when you're starting out, because looking at equations is gonna demotivate you, right? So, what I instead implore people to do is just look for applications of deep learning, right? So, you know, pick up YOLO-NAS and run, run it on some image, see the power of it. Open up, you know, ChatGPT or any of the other language models and start playing with it, like interacting with it. Start interacting with models, trying to build something with it, trying to do cool stuff with it, right?

01:08:01 You know, open up, learn some LangChain, and see what you can build, right? Just see the magic happen, get inspired. Then once you're kind of inspired, right, if you think it's cool, right, some people probably won't think it's cool, they'll just, you know, be like, okay, cool, whatever. That's fine. But if you think it's cool and you're interested, then dig a little bit deeper. And digging deeper, there's like a couple of places I recommend one of them: Andrew Glassner has this deep learning crash course. It's like three and a half hours long. But it gives you just proper intuition for how, how all this works. Very good return on time investment. So, like he wrote this book, the Deep Learning Illustrated Guide, huge, huge, massive book, right. No math-

Jon Krohn: 01:08:48 Deep Learning: A Visual Approach.

Harpreet Sahota: 01:08:49 Oh yes. Deep Learning: A Visual Approach. Yes, Deep Learning Illustrated is another book I'm about to talk about. Deep Learning Illustrated Approach. Great book. And then once you do that start learning some PyTorch, right. Just, you need to move away from SciKit-Learn. Going from SciKit-Learn to PyTorch is a bit of a mental shift. But you know, Daniel Bourke, I'm not sure if you've interviewed him on your podcast or not, he's awesome. He's based out of Australia. Highly recommend him. @mrdburke on Twitter. But he's got this Zero to Mastery PyTorch course. Go through that because you're gonna get a bit of intuition about what's happening under the hood. Then you're getting your fingers on the keyboard, you're getting your hands dirty, you're coding, right? This is nice because it's completely self-paced. And you're gonna learn how to code in PyTorch, which I think is the best for deep learning in about a week, right? You'll be comfortable with PyTorch in a week.

01:09:43 So, now once you got that, then you pick up this book by Jon Krohn called Deep Learning Illustrated. And this will get you more to the math, right? So, get more into the math with this book than jump on YouTube that teaches you the basics of deep learning math, you know, two, two and a half hour time investment and you're learning from an Oxford PhD. Come on, that's amazing. And then, and then and then I think one of the most important things, just you have to understand backpropagation. I think once you've gotten to this point, right, just spend some time understanding backpropagation. Just make sure you kind of understand what you know intuitively what's happening. And, you know, maybe a little bit mathematically. I like Andrej Karpathy's, I think it's called Neural Network Zero to Hero on YouTube. Great, great resource for that. You actually end up building you end up building like a mini version of PyTorch. I think he calls it like minigrad or something like that. But it's amazing. It's great.

01:10:47 Once you've done that, then get an understanding of more foundational architectures. You could, you know, once my LinkedIn learning course is out, go through that, go through some of the foundational computer vision architectures. Yannic Kilcher has a great YouTube series on classical papers. It breaks it down in an easy-to-understand manner. And then just join some community, you know, be around other people who are into the same stuff. You want to be around people who have a broad range of experience from learners to experts. And then finally just projects. Just do projects, get on Kaggle, do a bunch of projects. I think that's the best way to go about this.

Jon Krohn: 01:11:29 Nice. That was a great roadmap there. And yeah, for people getting into deep learning, that sounds like a really great flow from Andrew Glassner's Deep Learning - A Visual Approach. I didn't know that he really, I guess he doesn't have very much math at all in there. And I haven't read his book, but you can confirm.

Harpreet Sahota: 01:11:46 Yeah, not too much math. It's, it's all like, it's just a, you're understanding the concepts through illustrations, which is amazing. Yeah.

Jon Krohn: 01:11:54 Nice. Yeah, and it's interesting with my Deep Learning Illustrated book, and thank you very much for the shout-out there. I set out to have it be as unmathy as possible, but it's interesting, but it does have some math certainly. And so it's, it's cool to me that somebody else has come up with an approach that is even more visual. So, the people who want that completely visual approach can do Andrew Glassner first, and then, yeah, I've got some of the math in my Deep Learning Illustrated book. After I wrote Deep Learning Illustrated, I realized that a big gap is people's ability to apprehend the underlying linear algebra and calculus of deep learning, like the calculus associated with backprop. So, since then, since I wrote Deep Learning Illustrated, a lot of my content creation has been around these foundational subjects. Completely different from the idea of foundational models.

01:12:49 Where yeah, these foundation models are, yeah, these, the huge models that you were describing earlier, foundational concepts completely different. I'm just talking about like math and computer science and probability and statistics, that these foundational subjects you need to know to be able to understand machine learning well. And so I would, I I'm, I put a lot of time into and I think and based on feedback, I can confidently recommend if you don't understand exactly how backprop works today, I have no doubt that Andrej Karpathy's resource is great. All of Andrej Karpathy's resources are awesome, but I have a calculus for machine learning YouTube playlist that goes from and maybe, you know, if you already know calculus well you can skip some of the beginning videos where I show you how calculus works. So, I explain in an intuitive way and with lots of hands-on code demos, how calculus works, how partial derivative calculus in particular works, and then how we can use partial derivative calculus to do backpropagation.

01:13:50 And it has, and so my guess is that it's a longer journey than Andrej Karpathy's. Because I think it's something like seven hours of videos. And then if you do the exercises as well, you're looking at even more time invested there. Because I give you like exercises and solutions at different points, checkpoints throughout this curriculum. But yeah, whether you are yeah, wanting to get just get started on the underlying calculus to understand backprop or you want to jump to later videos and get deep into the weeds on how backprop works using calculus principles and do it in a hands-on Python-based way. Yeah, I think I I it's my own resource, but I think it's good.

Harpreet Sahota: 01:14:32 I've linked to, linked to that resource many times, like it's a great YouTube course. And another kind of interesting resource I like is there's like this a series of manga books that touch on a wide range of topics. I've got the entire set, but there's a book there on calculus and on linear algebra and they're like, you know, proper like comic books, but it teaches you calculus and algebra.

Jon Krohn: 01:14:58 Oh, manga. Oh really?

Harpreet Sahota:	01:15:00	Yeah, yeah, yeah. The Manga Guide to Calculus and the Manga Guide to Linear Algebra. Super good.
Jon Krohn:	01:15:06	Awesome. So, near the end of every episode I ask people for book recommendations, but you have just given us a ton. So, I think we've covered that question, unless you have any other books you'd like to add.
Harpreet Sahota:	01:15:17	You know, I used to, I used to, I have, I've traded, when I was recording the Artist of Data Science podcast, I read a lot of books like, cause I had so many authors on and since I kind of put the podcast on hold for now, I spent most of my time reading research papers in the morning whenever I have free mornings. I have not read a book in like six months, sadly. But the one that I have currently just gone back to rereading is Deep Work by Cal Newport. I think that's a good book. Important book for people who are in roles like ours that are knowledge-intensive and require a lot of thinking.
Jon Krohn:	01:15:55	Yeah. So, important to be able to carve out a little bit of time every day to be able to work deeply. It's absolutely essential. All right, Harpreetet, so, awesome episode today. Thank you so much for taking the time. And hopefully you can even consider this to be some of your deep work for the day. I actually usually do, otherwise, if I didn't include filming podcast episodes, I'd have an embarrassingly small amount of deep work done every day. So if people want to be gleaning amazing insights from you after this episode, obviously we know that your Deep Learning Daily podcast is something that they can be checking out. What other, what other ways should people be following you?
Harpreet Sahota:	01:16:38	Nowadays I'm mostly on Twitter so @datascienceharp on Twitter, so find me there. You know, I still have, I still have like a huge following on LinkedIn. I'm just not as active on LinkedIn, just because the algorithm has been

unfair to me far too much. But Twitter's just like a cool, like if you're, if you're wanting to get into deep learning and keep up on trends and on research papers and things like that, like Twitter's the place to be. LinkedIn I find is more classical ML data analyst, data engineering, business-kind of focused. But Twitter's where all like the cool stuff is that, that I'm into at the moment. So, find me there.

Jon Krohn: 01:17:16 Nice. So we'll be sure to include those links in the show notes. Harpreett, thanks again for taking the time and maybe in a couple years we'll be catching up with you again.

Harpreet Sahota: 01:17:27 Absolutely, man. I'm looking forward to it. Who knows, who knows where I'll be in a couple years, but you know, I'll always be happy to come back on.

Jon Krohn: 01:17:34 Nice, catch you in a bit. Harpreet.

Harpreet Sahota: 01:17:36 Cheers.

Jon Krohn: 01:17:42 Nice. So, great to catch up with Harpreett on air. He's clearly flourishing in his Deep Learning Developer Relations Manager role and making a big impact. In today's episode. Harpreet filled us in on how object detection is a machine vision task that involves drawing bounding boxes around objects in an image and then classifying each of those objects. Talked about how object detection models have become much faster in recent years by requiring only a single pass over the image, as with the renowned, You Only Look Once YOLO series of model architectures. He talked about how Deci leveraged their AutoNAC neural architecture search to converge on YOLO-NAS, an optimal architecture for both object detection accuracy and speed. He talked about how hybrid quantization selectively quantizes parts of a model architecture in order to increase inference speed without adversely impacting accuracy and how the future of AI may lie at the intersection of the Internet of Things and smaller generative models.

01:18:37 As always, you can get all the show notes including the transcript for this episode, the video recording, any materials mentioned on the show, the URLs for Harpreet's social media profiles, as well as my own social media profiles at superdatascience.com/693. That's super datascience.com/693. If you live in the New York area and you would like to engage with me more than just online, you'd like to engage in person, on July 14th, I'll be filming a SuperDataScience episode live on stage at the New York R Conference. My guest will be Chris Wiggins, who's Chief Data Scientist at The New York Times as well as a faculty member at Columbia University focused on applications of machine learning to computational biology. So, not only can we meet and enjoy a beer together, but you can also contribute to an episode of this podcast directly by asking Professor Wiggins your burning questions on stage.

01:19:25 All right, thanks to my colleagues at Nebula for supporting me while I create content like this SuperDataScience episode for you. And thanks of course to Ivana, Mario, Natalie, Serg, Sylvia, Zara, and Kirill on the SuperDataScience team for producing another eye-opening episode for us today. For enabling that super team to create this free podcast for you, we're of course, deeply grateful to our sponsors. Please consider supporting the show by checking out our sponsors' links, which you can find in the show notes. Finally, thanks of course to you for listening. I'm so grateful to have you tuning in and I hope I can continue to make episodes you love for years and years to come. Well, until next time, my friend, keep on rocking it out there and I'm looking forward to enjoying another round of the SuperDataScience podcast with you very soon.