EPISODE 1375

[INTRODUCTION]

[0:00:00.3] KP: Machine learning models must first be trained. That training results in a model which must be serialized or packaged up in some way as a deployment artifact. A popular deployment path is using TensorFlow.js to take advantage of the portability of JavaScript, allowing your model to be run on a web server, or the client. Gant Laborde is Chief Innovation Officer at Infinite Red, a React Native consulting team, and the author of *Learning TensorFlow.js: Powerful Machine Learning in JavaScript* from O'Reilly. In this interview, we explore the use cases for TensorFlow.js. Gant, welcome to Software Engineering Daily.

[00:00:42] GL: Thank you. Good to have me.

[00:00:46] KP: To kick off, can you tell me a little bit about how you got started as a developer?

[00:00:50] GL: Oh, well. I can tell you, my family was – Well, my dad was anti-computers 100000%. Of course, I was very interested. I would play video games. I had an old Nintendo system. Finally, work made him have a computer. I started messing around on it. I probably broke it three or four times. Then one day, I got it to say, “Hello, Gant. You are writing Quick Basic.” That was it. That was it. I was into programming from that day forward, using the help file to learn how to code.

My dad said, “You have to wrestle.” If you ever seen me, I don't – You're like, “Oh, look at this guy.” I had to wrestle in order to get my first computer for Christmas. I got that computer. Went to college for computer science. I don't recommend it. It was a lot of time and energy for me to just come out and do all the fun, pragmatic things that I'm doing now. I had that computer all through college as well. It was my high school wrestling computer had to last me. It was a Pentium, running Windows 98 for the longest time.

Then I got into just building cool stuff for people. I really enjoyed that. That pushed me through the college career and into my professional career. Oddly enough, I think I'm doing the same

thing now. I'm just building cool stuff for us, and for a bunch of clients looking for someone who's willing to jump into R&D and figure out how something works.

[00:02:30] KP: When did you found Infinite Red? What do you guys get up to there?

[00:02:35] GL: Funny story. The CEO of Infinite Red, Todd Werth, went ahead and found me. Here in New Orleans, we don't have a lot of tech. It was a little bit difficult at first. I would build a website, I didn't have anybody to show it to. Then fortunately, after a while, through the open source community, and through conferences around the world, I wound up meeting some pretty intelligent and awesome people. I started doing consulting. I wrote my first book on Ruby Motion back in 2014.

I was doing consulting all across the US. I was consulting in Florida, North Carolina, in California. Todd was running a company called Infinite Red in California. I found, I just liked one of my clients. I just loved consulting for Infinite Red. It was my favorite thing to do. That was treated well. Then after a while, Todd went ahead and merged his company with ClearSight, with Jamon in Portland. The company got bigger.

It got to a point where the company was really turning into this dream job, this dream company that I always wanted to be a part of. That's honestly, what the initiative was. What's a company that you would want to work at? It was there their mission statement to start it off. Finally, in 2017, I came to them and I said, "Look, I think I've hit the max of where I'm growing." They said, "We see that, too." I said, "I want to buy in and be a member of this company," and so I have. Since 2017, I bought in and I'm one of the three owners of Infinite Red. We still just have a great time finding intelligent, smart people and building mobile apps and building cool stuff.

[00:04:22] KP: What's changed about mobile apps in the times you guys have been doing that?

[00:04:27] GL: That time, used to be that you'd have to watch all the keynotes to keep up with everything. Because in the beginning, people would show up and say, "We need an iOS app. If we have extra money, we'll get an Android app." That has changed significantly. People show up and say, "We need an iOS and Android app. And, we want our team to manage that code, and

we find it to be pretty much reusing that. If you can reuse some web code as well, that'd be fantastic."

Which was an impossible order in 2014. We use React Native, so we're in a React-based framework. we get to actually – we have some websites that I deliver a significant portion of reusable code amongst web, Android, and iOS. JavaScript as a top layer on top, like the C on top of assembly. React Native is that JavaScript layer on top of those native platforms, helping us actually deliver all this cool stuff.

[00:05:28] KP: Having done that successfully a bunch of times, you've seen the full cycle of building a React Native app. I'm aware of the concepts. I've messed around with it, but never enough where I feel like I know the gotchas. Are there any gotchas out there in the process?

[00:05:43] GL: Oh, yeah. Absolutely. The gotchas aren't where people probably would tell you they are. The one key aspect here is that you have this bridge that you're going across, when you're telling JavaScript to tell Native what to do. It does whatever you tell it to do. If you decide to completely flood that bridge, that's you. You did that to yourself, that was your fault. We've gotten really good about pushing stuff down to the native in a very effective and efficient way. The other thing that comes in, I'll say, as a gotcha is, it's very easy to overshare styles in code. I think that each platform has an independent feel that you need to know when to break unity between the code and show it just a little bit different on iOS than you would on Android, so it's not like a WebView that's shared. You wouldn't have that. It's actually native code that feels native on one platform versus another.

I think that that's something that takes a little bit of time. Because you can't go bring the same interface everywhere you go. It's very important, if I were to do a Apple TV app, the interface that the person's interacting with is very different. I could write that in React Native, and it could feel very unique for that platform, and then still share a lot of the business logic with an iOS app. Knowing when to do that, knowing when to break the rules, that's the hard part.

[00:07:13] KP: Makes sense. A lot of people strive to be full stack, but it's somewhat uncommon that you know React and React Native and also, can write a book about TensorFlow. How did you get versed in these two, not necessarily neighboring technologies?

[00:07:28] GL: Well, I could tell you believe it or not, that path did go through React Native. I am the CIO at Infinite Red, which is Chief Innovation Officer. There was that episode of Silicon Valley that came out where they did hotdog, not hotdog. If you're a fan of the show, you know exactly what episode I'm talking about.

That app actually exists. You can download it tomorrow, or right now as you're listening to it. It completely exists. You could take a picture of something, and it'll tell you if it's a hotdog, or not a hotdog. When I found out that that actually was a real app written in React Native, I said, “Wait a second. I know everything there is to know about React Native. How do you write an app that can actually look at a photo and tell you what's in it?” This is what? 2016, 2017 that I saw this.

Believe it or not, it was really cool. It opened my eyes to computer vision. That showed me what cool things that are coming onto the horizon that we should all be figuring out how to do, that are going to be features and all kinds of cool apps. I’ve started digging in deep, taking Coursera courses, learning everything I can, going back to my linear algebra days, and then ultimately, finding out how to do computer vision and how crazy the AI revolution actually is, and all the new features and all the new things that are about to hit that we're going to have to deliver on mobile apps.

It's like back when we used to deliver just iOS apps, and now we have to deliver on all these platforms. I see that again, coming in as people are going to come in with an AI idea, and then they're going to want AI everywhere inside of their stuff. That led me into a multi-year dive into learning everything possible about TensorFlow and machine learning, and ultimately, writing the O'Reilly book, *Learning TensorFlow.js*.

[00:09:24] KP: Well, sticking with the link there between the mobile app and computer vision, I'm wondering if you could walk through, I don't know if there are best practices, or if you have just a single vision for how to do it. Do I train my model and deploy some artifact? Or what does the actual development process look like to get that going?

[00:09:41] GL: Well, it's wildly unique for someone who's a developer to even conceptualize the idea of data-driven development. Because the way I liken it is to, if you're playing a video game,

if a person coded the AI in that video game previously, what they did was they coded rules. They coded instructions, and they took abstract concepts as best as they could, and put them into this codified form. Whereas, here's the difference for people who are new to this, and I don't want to pass over that. You said the word ‘model’, which funny enough, means something different if you're using Blender versus video production, versus math, versus AI.

It's basically, a model is just a function, but the function is being trained not by a programmer. It's being trained by data. You give it a 100 examples of a hotdog, and you give it a 100 examples of a bird and you say, which one of these is which. I don't actually code how to tell the difference. I let it see those examples almost like flashcards and figure that out. That's just a mind-blowing moment for most developers right off the bat.

Then, you can take that a step further. In 2018, I did a talk on whether someone is, or is not Nicolas Cage. It was a really fun talk. What we did was, I had to pull down hundreds and hundreds of photos of Nicolas Cage. Then, I had to grab other people who look like Nicolas Cage, and then show the AI these photos, until it could actually distinguish him from anyone else. Then for the talk I did at React Native View in 2018, we had someone hide in the audience with a Nicolas Cage mask on. We use the AI on the phone to identify him in the audience and bring them up and then retrieve the Declaration of Independence.

[00:11:42] KP: Well, JavaScript isn't historically known as one of the main languages for machine learning. I'm thinking of Python, maybe R, a few other things. What gave you the idea that a whole book would be the right path to lead someone to be doing a mail in JavaScript?

[00:11:56] GL: Yeah. The interesting thing that comes in here is that don't ever quote me on this, but the truth is, JavaScript is not my favorite language. It is the most popular language. It is the most powerful and ubiquitous language that we have there. My heart still deeply belongs to Ruby from a long time ago. Love you, Matts. Great programming language. I wish it were everywhere. Instead, we have JavaScript.

JavaScript can do amazing things. Often, it's almost placed as a toy language by people who see it go to things like drones, or see it go to AI. JavaScript doesn't play around when it shows up. You've seen websites that have full access to your web GL, web GPU, web assembly, being

able to really tear up the speed on a computer to do these things. The truth is, when JavaScript and AI meet, there's this unlocking moment that you can't get from Python. Perhaps, you'll take a look at a Python website that's got a really cool model that will turn you into an anime character. I think, there's selfie2anime.com.

If all of us hit that website right now at the same time, we'll get a message saying, “Sorry, there's too much traffic. Come back later. We can't process all these images at the same time.” Well, JavaScript can run on the server, and it can run directly in the browser. If I actually put the model on your browser, you can hit the website whenever you want. I can actually use your machine, your GPU to perform the inference for that model, and do the generation of that model directly on your machine and not hit the server at all.

It's not a play language. It is serious. It's here to show up and kick butt. As AI is going to a bunch of edge devices, like phones, like Raspberry Pis, like small computers, it's going to be really, really impressive to stay with JavaScript, because as you know the rule is I think, it's Atwoods Law, if it could support JavaScript, it will.

[00:14:07] KP: What are some of the benefits of running my model at the edge on the client hardware like that?

[00:14:12] GL: Oh, its perfect example would be – I guess, if your model is not secret sauce, if it's – it's a little bit of a novelty now. Selfie2anime was the example a second ago. It's just impressive that the bear is dancing. To quote, “The inmates are running the asylum.” We're impressed with the bear is dancing, not how well the bear actually dances. Then that wears off, and now we actually care how well things actually perform. I'll say that if I'm able to put my model on your machine, and it's directly applying hair dye, or makeup to your face, and then tells you exactly what hair dye to buy, and what makeup to buy to have that exact look on your face and it's all in your cart, you just hit purchase, then I don't care about the secret sauce anymore. I'm not sending it to the server.

I'm accessing your webcam. I'm giving you the model. I'm putting it all on your machine, because now, the cool part is not that it's possible, but that you're getting these right now. You hit the purchase button, and I make a sale, because my website showed you that hair color and

that makeup together, and the other person's got to wait and send the stuff off to the server, or doesn't even know what it would actually look like in person, I'm passing you up using AI. Then the truth is, that's going to be in products everywhere, pretty fast.

[00:15:37] KP: Are there any drawbacks? Obviously, there's the network latency you have to take on when you're going to do something on the server. I don't know, is there a cold start problem if I have a really big model that's memory intensive, or anything like that to consider?

[00:15:50] GL: Well, if you're going to be putting it on the actual device, there's sometimes these models can be big, especially for the web world. If I had a 25-megabyte model, I really need to depend on my UX engineers to give me a process to give them – Have you ever heard of the elevator mirror effect? Elevator mirrors actually give you multiple benefits, and that's why you always have mirrors in elevators. One of them being, the passage of time while you're looking at yourself. Additionally, me feeling less claustrophobic and also, keeps people behaving.

There's all these potential needs for these large files, which web is all about precise, small files. How many kilobytes can I get this? How many bytes can I get this? Then I show up, I'm like, “Here's a 25-megabyte file we need to put on the client.” This is where design and UX and storytellers are really going to benefit and be able to do a lot. I will say that it doesn't come at zero cost.

Then also, depending on the person's machine, if there's no GPU, if it's an older machine, there's the chance that some of these models are just too advanced for the machines right now, until we start to see actual AI specific hardware come in to most devices.

[00:17:08] KP: You'd mentioned the concern about your model being private. Maybe you don't want to get out or something like that. Well, I do respect that concern. It is true, someone could grab it out of their local cache or something. Do you find that that's a concern for most enterprise companies?

[00:17:23] GL: That's going to be a concern for people who are specifically in a data science position. If I have a model that's tracking 80 points on your face, and that I've spent millions of

dollars of research with data scientists to go ahead and get those 80 points on your face, then I need to be extra careful, because that's coming back to the secret sauce that's actually selling what's going on there. Then we have something out there that does give 30 points on a face.

If I'm really careful, and I can get something that's significantly refined, I have to be really careful pushing anything to edge devices. That includes mobile apps. That includes all kinds of things. We start to see this problem, again, of reverse engineering to steal, have someone solving a problem that we haven't seen since the 90s and early 2000s, where people really cared how a function actually works, and coming back and stealing how problems are solved. It's cool. I think, it happens with every revolution.

You're right. There are already things in place to make browsers enforce secured files to the neck and mark something as secure. Then, it would stop the honest thieves from quickly downloading it.

[00:18:41] KP: Very cool. Well, I've deployed some machine learning models in my life. when I've done that my collaboration has generally been with someone that has a title like, DevOps, or back-end engineer. I'm handing it off to a technical, behind the scenes architect, I guess. That person generally has a very different skill set than a front-end engineer. Obviously, some overlap. Do you envision a transition here happening where a machine learning person will instead be collaborating directly with the front end?

[00:19:08] GL: Yeah, I do. I see that there's this pragmatic section of getting people to develop a model aren't the pople who actually want to put it in production and check the performance and verify all the devices, as well as push updates to it at all. The truth is, where we're going with this data this, data is so valuable. It's been referred to as the oil of AI. It's going to be very essential to inroad new ways to get that information back. Not only to get that out there in an effective way that new model updates can roll out and new AI can roll out. Because I believe iOS would beat everybody with Siri and voice control, but they tied everything to the version of iOS.

Every time they wanted to roll out a new Siri, they would have to roll out a new OS version, which was much more difficult, rather than just doing this simpler update. Having models update

is itself an amazing, cool problem that we see already inside the developer world. I have a new version of this app. How do I do a new rollout of that? I have a new version of this web page. How do we do that? It'll come in, I have a new version of the model, how do I go ahead and get in and put that?

It's also a bi-directional problem, because as we figure this out, as people use the model, and the progress of the model is important, we'll need metrics, and then we'll need reports back that do not violate the anonymous, or the personal privilege of the people to bring their information in, and maybe their faces, because we're seeing that already. Some apps send people's faces to a server and everybody loses their minds, because that's a terrible thing to do. How do we anonymize the information that's been processed by the AI, that's a very successful AI and bring it back so that we can still improve our process on going in more of a agile form? That'll be federated learning and all kinds of cool stuff. Yeah, absolutely. I'd say over the next 10 years, we're going to see a bunch of jobs spring up specifically with that.

[00:21:16] KP: Very cool. On the topic of deployments, I like your hint that maybe machine learning people can take a lot of notes and lessons learned from the maturity of traditional software development. There have been a lot of good best practices put in place. Yet at the same time, if you deploy just the code you've written for your app, it works, or it doesn't. You either hit a bug, or you don't. I don't necessarily know that that's the case in machine learning, You could have situations, like the novelty effect coming into place. Do you see any extra challenges, or have advice for people who are rolling out models and want to be careful about it?

[00:21:53] GL: Yeah. I believe that the people don't understand the problems as well right now that come out with that. You're right, like when we started this conversation, we talked about what's different in programming a function by hand and programming function by data. There was a moment of learning. I think that when you get into the middle of it, where you have it, a function usually says, “This is true. There is a bird.” Or, “This is false. There is no bird.”

There's another gotcha there for all of you, developers, which is it's a percentage accuracy. It never says, “I see a bird.” It says, “I am 72% sure that this is the primary class of what this is, or I'm 75% sure that this is a bird. “You have to choose what threshold is going to be the correct

threshold there. Then here, coming back even further in the future of being able to roll out new models that might actually have different tensor outputs. They might actually have different feedback.

Then, they have issues. Or maybe, you might even roll out something with bias in it. There's going to be an important process there that says, that you're not used to, which is going to be specifically, how do you find out and how do you get that back? The further we go out into the future, the more cloudy it gets. I think, all these problems are the problems that we're going to be paying people to solve.

[00:23:23] KP: Well, I suspect most listeners, most software engineers in general, will know of TensorFlow and what it does, even if they're not a user or an expert in it. They might not be aware of TensorFlow.js. What is that distinction?

[00:23:37] GL: TensorFlow is Google's answer of a framework specifically to do machine learning. It's a machine learning framework provided by Google. Now, what ultimately happens there is that was built in the classic Python mentality of, we have our mega servers, we have our server farm, we have all these other pieces that we can work together to solve this very complicated problem. When we took those giant server farms, and we tried to put them on a phone, we found that even though the model size is smaller, there's a big problem there that needs to be solved.

You had to pay attention to what does a mobile device do effectively, and how do you optimize for that? There's the idea of saving models to a TensorFlow-like model? Well, then ultimately, there's this problem of how do we go to web? What's great about this is that TensorFlow has this large library of framework, and it's very Pythonesque on how it's all done.

When we're going to bring it to web, we're not going to shove Python into web. We're going to use JavaScript, but the same functions are there; it's trying to match parody of those functions to the Python version. Just expect camel case, instead of snake case. All the things that you learn when you learn TensorFlow should be able to apply directly to TensorFlow.js. All the things that you learn in TensorFlow.js are most assuredly, except for obviously, the parts that are very web-centric, will apply over to TensorFlow.

This idea of the framework being able to access shaders and GPU on websites, it's specifically a web-driven, mostly TensorFlow version. Of course, it can run on node and it can run on React Native, and it can run on a whole myriad of devices as well, because it's JavaScript. It goes where JavaScript goes.

[00:25:42] KP: Well, my instincts, maybe this is just some bias I have, but I think about going into Python, doing all my training, coming up with my model, and then the next leg of the relay race, I want to move it into TensorFlow.js. Is that a common pattern? Or are people doing machine learning and the training exercise as well in JavaScript more and more?

[00:26:02] GL: That is a great question. You can do everything you want, and this is the benefit of the multiple frameworks. You can do all your training in TensorFlow, and as long as it's supported by TensorFlow.js. You haven't done anything that just came out in the latest TensorFlow. It hasn't been implemented in TensorFlow.js. You could then convert your TensorFlow model to a TensorFlow.js model, just like you could have converted it to a TF lite model, and put it on a phone that way.

You convert it over. Now, you can actually just write the inference, or the implementation of the model on a website. Actually, that's what I wound up doing for quite a few of my projects, was doing the training side in classic Python, but then bringing it over and doing the actual, just implementation to show off in JavaScript, which goes wide very quickly. It's easy to share. We have the silly web. It's easy to put things out there like that, and have tons of silly examples, but a lot of them were actually created specifically in Python beforehand.

However, I will say this, node in some aspects is outperforming Python in training. It doesn't have the large library of data science backing it over a long period of time, but we're seeing new things, like Danfo.js, and new models from Google and Hal9.com, where we're actually seeing people really push into what if you wanted to do all your data science in the JavaScript style, and then go straight to JavaScript, where almost, we'll be looking at implementing really cool training websites, and then exporting the model, trying to go back to Python, trying to go back to original TensorFlow, go to TensorFlow lite, and originally starting off with TensorFlow.js. I'll say

that that's a new movement, but there's no cost in specifically going into one of the classics for a particular reason and then converting.

[00:28:01] KP: With regard to the book, what prerequisites, or background knowledge should I have before picking it up?

[00:28:08] GL: I wrote it for two audiences. One, your web developer who knows nothing about AI, or two, your TensorFlow expert who needs to go ahead and move their AI project over to TensorFlow.js. There's a little bit of redundancy, or simplicity, but we had a bunch of test readers; people who have never done AI before and people who've never done web before.

Both those audiences were very happy, and the success, because they felt like they were getting – the AI developers thought that they were getting a good, pragmatic way to see the benefits of TensorFlow.js, how to test it, how to implement, how to send it out, how to check it. The web developers were learning a good bit about what the hell is AI and what's going on here? That initial splashdown experience.

I'll say, if I had to put numbers on it, I'd say, 70% of the book is helping web developers get into AI. 30% of it is helping AI developers really understand web. You can slide that number around, depending on what your experience is.

[00:29:19] KP: Well, let's put ourselves in the shoes of that web developer. Let's assume they're accomplished in their craft, but all this AI, ML stuff is a bit mystical to them. If you just start looking up deep learning, you're going to read about linear algebra concepts and this thing. How much of that do you really need if you just want to have the hackers mentality here?

[00:29:40] GL: Zero. I wrote a wonderful article for Hacker News that has – it's one of my most successful articles I've ever written. It's called Machine Learning, Zero to Hero. That article did phenomenal. The PhD showed up and they got a little upset, because going from zero to hero, pretty much insults eight years of school for them, or something like that. They're not very happy with that idea, so they poked holes.

Immediately, people showed up and said, “That's not the purpose here. That's not what he's saying.” Honestly, throngs of people showed up and helped identify what the premise of that was. What I'll say for the book here is that, no linear algebra. Now, if you're afraid of math, you got to get over it. As a computer programmer, that happens at some point, at some point, always. You can get into it. You could do a workshop. You could say, I'm never going to do math. You have to just make your piece, understand math shows up everywhere. Embracing that means that you're going to be better at life. You're going to be better at programming, you’re going to be better at all these things. I don't ever hit anybody over the head with anything harder than basic calculations, like cosine, like basic trigonometry functions that you don't even have to understand.

[00:31:03] KP: Well, maybe we could zoom in on your John Cage example.

[00:31:06] GL: Oh, Nic Cage.

[00:31:08] KP: Yeah. John Cage would be totally different.

[00:31:11] GL: Johnny Cage, and I think that's a Mortal Kombat reference. Yeah.

[00:31:15] KP: Oh, John Cage, the composer was who I was going to say, the via Nic Cage.

[00:31:19] GL: I feel the culture difference between us, Kyle.

[00:31:26] KP: Well, let's take that as an example. What if I wanted to set out and do my own Cage example and build that up, and I didn't have the ML background? You mentioned grabbing photos. I could figure that out. I can do some web crawling. What does it take to actually wire it up? How much ML do I need to bootstrap myself?

[00:31:42] GL: For identifying the person's face, the first thing here is that there are already existing models. You have to understand what the AI is doing. There's a fantastic article by I think, fun machine learning, which is what is facial identification actually doing? What actually it does is it goes in and identifies key points on the face? Then for facial verification, there's this moment where you actually realize and normalized face, those key points on the face, despite

how much I contrast and contour my face and how many funny faces, even if I'm Jim Carrey, trying to make really fun faces.

Those average of all those points is somewhat uniquely identifying for me, that those would be at those distances from each other. Simply doing the AI gives you a set of points. Then you would just use basically, the distance formula, which we all know. Well, if you don't, then that's okay. There's lots of libraries out there already doing it. You do a distance formula for those points, and then that would give you a range of error to say that this person is person A versus person B, and then you can identify how much error those points can have.

That is a fantastic way to go ahead and train to get what the average points. If you forgot 200 photos of Nic Cage, and then you've got the average of all of his points and all of the crazy faces that he makes, that's going to be a unique identifier. That's a blog article, running a model and running a function away from creating a face verification process. It sounds super alien, because this is the first time you've ever thought to solve a problem that way, but that's not very hard.

[00:33:36] KP: Well, only 200 photos is very practical, but it seems too low. How can it work on so sparse a dataset really?

[00:33:43] GL: The model that was doing all the heavy lifting here was the facial data points. Being able to actually find those key points on the face. Training that model, that model was trained with hundreds and thousands of images, of tons and tons of different people. Those models were run through bias detection and all kinds of other problems. There's ones that people have trained themselves that are out there, and as well as Google, I think has quite a few blaze face, face match, a couple other things where they're giving key points on faces, that they've put their stamp on from Google Brain team. Those saw server farms and hundreds and hundreds of computers.

It's like, when we take a look at GPT-3, where someone can use this – if you've heard of it, they could ask it to write a story, and I can use the benefits of GPT-3, the actual use the function and the model now. But to train GPT-3 was estimated 12 million dollars' worth of work. The trick that I'm saying here is that for the facial verification example, is an excellent key point face model

that you can rely on, that will consistently find those key points on the face. That is the heavy, heavy, heavy lifting that we're getting from the data science community. Now, what's to stop us from making practical uses of it?

[00:35:14] KP: Well, I like the idea of, I guess, standing on the shoulders of giants in that way. In what other aspects can we take advantage of that with TensorFlow?

[00:35:23] GL: Great question. I have a really fun exercise I do in the book, where we train how to identify what house you would be in for Harry Potter in one chapter. If you draw a badger, you're going to Hufflepuff; a raven, you go to Ravenclaw; serpents for snake, and then a lion for Gryffindor. We train that on I want to say like, 10,000 images of drawings for those particular animals. That does a lot of the stuff there. Then what happens is we do this thing called transfer learning, which I'm sure you're familiar with, but not everybody is yet.

Transfer learning is taking an existing model, and re-appropriating it for a new purpose, using a lot of what it's already learned. What we do in the next chapter is take that and make it identify Star Trek insignias. Is this a Klingon symbol? Is this the Star Trek setup, or is this the Ferengi, is the three things it’s looking for? We do that with a really small data set of, I think, a total of 50 images total. Super small dataset, but we're standing on the shoulder of our own giant from the previous chapter.

Transfer learning is a great way to take models that do something and just change them a little bit. I think, it's like the StackOverflow of coding. If you find something that does almost exactly what you need, and then you modify it a little bit, so it works.

[00:36:54] KP: There's an intrinsic risk you take on in a project like that of what if for whatever reason, you just can't get it going? The model can't converge. Maybe it's not enough data, maybe the question is posed, and not the right way for ML. ML is a bit of a black box, when you deal with deep learning. Do you have any guiding principles for how you manage that risk, especially in light of doing it for potential client projects?

[00:37:18] GL: Absolutely. What you want to do is you want to identify how low you're going into the process, whether or not you've done it before, anybody's done it before. The lower you’re

going into it, the more you need to actually set aside an R&D budget. If you have our R&D budget, it's a little bit of an understanding what the client that this is going to tell you how close we can get with what you have. this is going to tell you what's going on here. The deliverable have a happy R&D budget is, this is where we're at, this is where I think we can get. This is just a meeting of the minds understanding that needs to happen there.

Whereas, if you were to use one of the services from Microsoft, or Amazon for facial recognition, and you're not even going to have to get down and dirty, and you're going to actually – you don't even need to go on the client side. You can actually do an API call and call one of these mega Fortune 500 companies to do it, then the risk is off. There's this saying, no one ever got fired for suggesting IBM. That's true. To the point, where I know someone told me a story recently that he had fighting IBM for a contract. IBM got the contract and sub-contracted it to him for his rate.

The person just went through IBM, paid the higher rate, but got the same work. This is just one of those things. The higher up you go into where these things have been done before and teaching yourself, knowing what's possible and what's out there is extremely valuable, because that helps you set the dial there saying, “No problem. We can put vague face verification in. No problem. We can identify what tools you have in your tool chest for you. Or we can make sure that this person, this car is not damaged.”

We've seen a ton of things, and there's a lot availability and high, high accuracy for that problem. When you start saying, “We want to do that at this level, I want to see if the cans in the store are damaged, or something new that we've never seen before.” Well, it's important that you actually you build in the expectation and you build an R&D budget. Now, if you're a nerd like me, that's your favorite thing.

[00:39:34] KP: Well, the opportunity to deploy models through JavaScript opens the gateway for a lot of stuff, in my opinion. Model deployment has historically been one of the harder things for enterprise companies to accomplish. That leads me to believe we're at the early days of potentially, a Cambrian explosion of interesting deployments. Have you seen, or are you anticipating any cool innovations that are going to arrive soon?

[00:39:58] GL: Yes, absolutely. I watch this all the time. This is my favorite show. What's coming out and what people are doing? I have a website, ai-fyi.com, where it's basically just, you're getting everything that I've found in a newsletter that I've seen, and it's getting easier and easier to run my newsletter. It started off with really digging into papers and articles and finding videos.

If you just look at AI on YouTube, AI on TikTok. I don't have Instagram, but I guarantee, it's probably blown up there as well. If you just see these videos coming out more and more, people are not only publishing new papers at a blazing fast speed, but now people are making entire careers, just reading those papers, and then telling you what they say and giving you examples and showing that. Then new stuff is coming all the time.

The web world is where that's getting hotter and hotter. Because we love showing off what we made. It's easy. It’s a link away. I do see that there's this really cool thing called the TensorFlow.js demos that Jason Mayes does. If you watch the Made with TFJS hashtag, he actually has a monthly show, where he's just showing cool things that people are doing with TensorFlow.js.

[00:41:24] KP: Very cool. Well, the book winds up with a chapter, where students are readers. I guess, we'll build the Capstone Project could be a cool thing to show off. For someone who's thinking about picking this up and walking through that process, what are they going to be able to create at the end of it?

[00:41:40] GL: In this process, I wanted to make sure that you create something that's not the classic. Now, I'll say this, when you pick up a programming language, you write “Hello world.” When you pick up AI, you pretty much – there's the EMNIST data set, and there's the property housing data set from the 90s in Chicago. There's the classic Harry – I'm not used to building those. I don't think they're very exciting.

I put a lot of effort into making a really interesting Capstone. That's this. I'm a terrible artist. I can't draw. I always wanted to draw, but I suck at it. I can paint by numbers, in a sense. I can go to any of those teach you how to draw things and paint by numbers. What I wanted to do was get AI to tell me how to paint by numbers. I thought, it'd be really cool to make entire murals of

dice, like actual dice that you roll, so that you could have, whether or not to put the dice on a five or a one or a two. If you put a whole wall of dice together, it'll draw whatever picture it is.

I've always been impressed by people doing these kinds of things with post-it notes, and wood clippings, and all kinds of things. I'm not creative enough to do that myself. I just asked AI to go ahead and do it. The final chapter is writing the AI that really uses everything you've learned from the previous lessons to give you a photo, and then it turns that photo into what dice would be necessary in order to go ahead and create that photo. Then I did that, and I sent it off to a person at Google. I actually did the TensorFlow.js logo, as designed by my final chapter in TensorFlow.js and sent it off to someone at Google and he has it hanging on this wall.

[00:43:34] KP: Very cool.

[00:43:35] GL: It actually works pretty nice.

[00:43:37] KP: Well, remind listeners what the title of the book is.

[00:43:40] GL: It's the O'Reilly, *Learning TensorFlow.js* book, *Powerful Machine Learning in JavaScript*. It's written by Gant Laborde, that's me. Then the foreword is by Laurence Moroney, who's a fantastic teacher, author and general person at Google.

[00:43:57] KP: Gant, where can people follow you online?

[00:43:59] GL: You could go ahead and find me on Twitter at Gant Laborde. Then, also have gantlaborde.com, where I mention all the places I go to speak. We can possibly meet in person and nerd out about all cool things, AI.

[00:44:12] KP: Gant, thank you so much for coming on Software Engineering Daily.

[00:44:16] GL: Thank you for having me.

[END]