EPISODE 336

[INTRODUCTION]

[0:00:00.5] JM: Most popular music today uses a computer as the central instrument. A single musician is often selecting the instruments programming the drum loops, composing the melodies, and mixing the track to get the right overall atmosphere. With so many different things to do on each individual song, a popular musician often needs to simplify some area of their work. The result is that pop music today consists of many simple melodies without much chord progression.

Magenta is a project out of Google Brain to design algorithms that learn how to generate art and music. One goal of Magenta is to advance the state of the art in machine intelligence for music and art generation. Another goal is to build a community of artists, coders, and machine learning researchers who can collaborate with each other.

Engineers today are happy to outsource server management to cloud service providers. Similarly, a musician could use Magenta for creation of a melody so she can focus on other aspects of a song, such as instrumentation. I'm personally really looking forward to the day where I don't have to think about writing a melody, I can just have magenta do it for me and I can focus on other things that are harder to describe to a machine and are better suited for a human.

Doug Eck is the guest today. He's a research scientist at Google. In today's episode, we explore the Magenta project and the future of music.

Software Engineering Daily is having our third meet up, Wednesday, May 3rd at Galvanize in San Francisco, and the theme of this meet up is going to be fraud and risk in software. We're going to have some great food, some engaging speakers, a friendly intellectual atmosphere. To find out more, you can go to softwareengineeringdaily.com/meetup. We would love to get your feedback on Software Engineering Daily also. Please fill out the listener's survey. It's available at softwareengineeringdaily.com/survey.

Now, let's get to this episode.

[SPONSOR MESSAGE]

[0:00:00.5] JM: Artificial intelligence is dramatically evolving the way that our world works, and to make AI easier and faster, we need new kinds of hardware and software, which is why Intel acquired Nervana Systems and its platform for deep learning.

Intel Nervana is hiring engineers to help develop a full stack for AI from chip design to software frameworks. Go to softwareengineeringdaily.com/intel to apply for an opening on the team. To learn more about the company, check out the interviews that I've conducted with its engineers. Those are also available at softwareengineeringdaily.com/intel. Come build the future with Intel Nervana. Go to softwareengineeringdaily.com/intel to apply now.

[INTERVIEW]

[0:03:08.7] JM: Doug Eck is a research scientist at Google. Doug, welcome to Software Engineering Daily.

[0:03:12.7] DE: Hey, Jeff. Thanks.

[0:03:13.7] JM: At Google Brain, you work on art and music generation, and you said that your main qualification for this project is that in 2002 you failed to produce music using an LSTM. From what I have read about the history of deep learning, there was only a small number of researchers who were interested in deep learning until recently. Why were you pursuing deep learning in 2002?

[0:03:38.0] DE: A really good question. I wish I could say that I had this vision that models like long short-term memory, LSTM, would take over the world. Really, I thought it was a great model for understanding timing and dynamics and times series analysis. Imagine whether it's the stock market, or spoken word and language or music, these models are really good at picking up on what's going on in a long time series. I had the chance to do post-doctoral

research with a group that invested LSTM and I took the chance to live in Switzerland for three years.

[0:04:08.7] JM: It didn't work at that point. What were the important —

[0:04:13.2] DE: Part of that is something of a joke. We had some success in doing blues improvisation on the piano with the LSTM, but at the time those networks were really small and we didn't have a lot of training data, and it's hard. I do like to see those [inaudible 0:04:27.1]. It's fun.

[0:04:28.8] JM: Yeah. What have been the breakthroughs in deep learning since 2002 that have made this a feasible project now? Because you're working on very similar work today.

[0:04:37.7] DE: It's absolutely true. At least on the recurrent network side a couple of things had happened. I think, largely, we can split this into two things; hardware and software. On the hardware side, machines have gotten faster. It turns out with deep learning, that matters in a way that is transformative.

Small models just don't do the same thing that big models do, and it can make the models big enough and give them enough data. They actually start to generalize and perform in ways that are frankly surprising. I'm not sure anybody really understands exactly how these models are working, but we definitely know that it's partially due to the fact that we can run larger models and train them faster.

Yet, on the software side, on the modeling side, there's also been — There's been a lot of work done on the last 20 years. We understand more about how to construct these networks, how to train them, and it's a bit of a black art so to speak, but there's been a lot of work that's been shared in the research community over the last 20 years that has done a better job of training. For example, when I was working on this in my post-doc in 2002, we didn't have what's called ReLU sharp 01 unit that we could train, and that changes things to give you one technical detail.

[0:05:42.0] JM: To give more people color on what you're working today. Actually, color, no-pun-intended, Magenta is the project you worked on at Google Brain to create compelling art and

music with machine learning. One goal of Magenta is to advance the state of the art in machine intelligence for music and art generation. What are some of the scientific conclusions that you've arrived at while working on Magenta?

[0:06:10.1] DE: In terms of scientific conclusions, we're trying to — First, Magenta is really just getting underway. I think just recently in the last couple of weeks, we've published some papers that are of interest and are really moving the field, I hope. In terms of science, I think one thing we're realizing is as computation moves deeper into our lives and AI moves deeper into our lives, by that I mean something as simple as your cellphone gets smarter or maybe someday you'll have an ear bud that's smart that's giving you directions, or something like Google Glass, but smaller.

The challenge has moved to how can we get computation to work well with people. People matter, and they've always mattered, but they matter more on the ground. I think what we're looking at in Magenta can be recast as we're working in a domain where we don't know exactly whether what we're doing is right or wrong. That is; have we done a good job at painting a picture, or making music?

The only way we'll know is by asking users and getting their responses and measuring that learning from that. That loop of putting data out there, examples out there, measuring the response, understanding if people like it or hate it, and then improving is one instance of reinforcement learning. The same problem we're trying to solve with robots that navigate in open spaces. We're trying to make some strides there.

[0:07:22.6] JM: The other goal of Magenta is to build a community of artists and coders and machine learning researchers. How have you been getting these different groups together to talk and collaborate effectively?

[0:07:39.4] DE: Building community has proved to be one of the more challenging things we've seen in Magenta. In one sense, we've been very successful. Magenta is part of the TensorFLow GitHub which is one of the most popular machine learning GitHubs around right now. The Magenta repository in the TensorFlow GitHub, that counts for about 10% of that whole GitHub. We're followed and people are forking our code and people are working with it.

What we haven't seen and what is generally a challenge is specific creative coders coming along and working with artists and giving us poll request to improve what we're doing, we've seen people use Magenta and we've seen people improve Magenta, but we haven't seen the joint of that happening, if you get it.

We haven't seen some creative coder come along, make some art, say, "I love this." Say, "Oh! I want to do this differently with Magenta. Add more code and bring that back in." I think that responsibility lies on us to do a better job of having the right API and the right set of tools for users. Every month we're reinventing, and every month we're changing and we're hoping to get it right this time. If we don't get it right this time, we'll get it right next month.

[0:08:41.8] JM: I'm an engineer. I'm also a musician, and I think a lot about the collaborative workflow between what an engineer and a musician might be. You look at a typical company that has a mobile app and a backend system and maybe they've got an android app and an iOS app and there are some cleanly partitioned roles there.

Are you starting to think about what are the cleanly partitioned roles in the machine learning music teams of the future?

[0:09:20.6] DE: As you've been asking that question, I've been nodding in agreement in terms of what — This is the most important question I think we could discuss with respect to building out this kind of community. I can tell you that we have gotten it wrong in the past. That is the first release of Magenta, was built around the command line.

For your listeners who aren't familiar. TensorFlow is built on top of Python. It's actually a bunch of C++ code, but the user interacts within Python. We thought it would be great to have a bunch of Python command line tools that would be used to train a model and then generate some music from that model.

In retrospect, that's kind of silly. It's like, "Okay. What is a musician want to do?" They want to bring up a terminal and they want to run a Python prompt that then dumps a bunch of midi files

into a directory." That's not the workflow we wanted. It sorts of surprises that we were sort of that naïve about things.

This begs the question; what is the right — I think you put it perfectly. What's the right division of labor so to speak, or division of complexity between software and musicians. I think at the very least, we've decided at the most minimal level, Magenta is about being the glue between TensorFlow and the creative community. By that, I mean very specifically, concretely in terms of machine learning and engineering, not like philosophically, an API for shuttling midi data back and forth between a TensorFlow model that's trying to make some predictions and a user who's just playing on a keyboard.

We want a user of that keyboard to not think it — When I say keyboard, I mean a music keyboard, or a piano keyboard. We don't want that user to think about how to quickly, with low latency, move music data around so that a really cool model can do some inference on that data.

Then, at the same time, we also, I think, want to have really clean ways for uses to take audio data and for users to take midi data, which is a data that stored by an electronic keyboard that note onsets and offsets, and shuttle that data back and forth in models to train. That's pretty small. That's not asking a lot. Just to say we're going to be this simple API that is kind of responsible for data in and data out. Then, again, if you look at what machine learning is about, that's like kind of 75% or 80% of the work anyway. If we can do that well, we've already achieved something. Then, the question is; what else do we want to do? I think that's open.

[0:11:33.4] JM: When I was in college, I was writing a lot of music on the computer and I would look for friends and other people to work on music with partially because I have this nostalgic view of, "Oh! You're in college. You start a band. You get a drummer, or you get a keyboardist, you get a guitarist, and so on, and you collaborate together."

The thing is modern music, as you know, a lot of the workflow is just one person sitting at a laptop and there's not as much collaboration. We could talk about why there is not a GitHub for musicians, why there's not a digital audio workstation that works for multiple people together.

We could talk about why the most popular producers are singular producers, rather than teams of producers. Something doesn't work in the collaboration or electronic musicians.

What I like about the approach that you're articulating with Magenta is you're describing a software tool that would work for the singular laptop musician, because at this point as an engineer who is a hobbyist musician, I don't really like to collaborate with other people unless they're extremely reliable and they have some sort of skin in the game, which is frankly rare in the musical community. The musicians that collaborate most effectively in recent memory have been people that are paid for, like outsourcing the mixing and mastering to somebody else.

Where you have this, basically, an API contract with the musician that you're working with. To put it more concretely, I like to work with the software, and I would love to outsource things that I don't understand very well to pieces of software, for example; melody generation. For many people, they would say, "Oh! Making a melody is core to creating music,"

You talk to many electronic musicians today and they're like, "You know, the melody is not the hard part. It's the eight hours I spend on the soundscape and the synthesizer generation." You could either automate those things or you could just focus on those things and outsource the melody generation to Magenta. You're building a tool that actually makes sense for the workflow of the musician of the future.

[0:14:04.0] DE: First, if we have future chances to do podcast, can you go in my place and talk about what Magenta is, because you really have a better job than I am with what we're trying to do.

A couple of comments on that. First, I think you nailed it in terms of the isolation of doing electronic work. However we want to put it; EDM, or just work on a DAW. One way to look at that is to recall that this might be more akin to music composition and music performance if we look back 40 years. I think composers have always worked alone, or almost always worked alone. There are pairs of composers that you'll see out there, but composing is never really been kind of a traditional group thing.

One way to think about your work at a DAW if you're sitting in Ableton or something like that is, really, what you're doing is composing. It just turns out you have this amazing tool to be able to listen to the music you're composing as you're making it. That carries quite nicely.

I think you also nailed it that one reason we're not using these same tools for performance in the same is that they just don't work very well for performance. It's really hard. I've actually seen performances. There are groups that take laptops and perform on stage. They're very researchy groups. There's SLOrk and PLOrk. These are real names. The Stanford Laptop Orchestra, and the Princeton Laptop Orchestra. They actually need specialized software that synchronizes the timing of their laptops o they can play together.

In fact, that computer programming language is called ChucK. It was done by a Ph.D. student at Princeton named Ge Wang who is now a professor at Stanford. Which points to the fact that it's such a hard problem that someone does their Ph.D. to build a programming language that has the timing characteristics such that you can link together a bunch of laptops and even then you're in the same room so that you can make music together.

Which makes, for me, like a fascinating problem and a fascinating direction to go in. It's not really one that we're pursuing actively in Magenta parsley because I think ChucK is already cool. That's one thing.

The other thing that I would respond to is I love the idea that you can maybe ask a tool to do something as basic as write a melody for you. In one framework, that melody is the thing that you're trying to produce creatively. It's interesting to think that if you let something like Magenta provide you with melodies, it's not that you're stopping being creative, it's just that you're getting to off source that. You put it perfectly. That allows you to do something else.

You certainly see this in almost all sort of Ableton style music where a loop is a thing and you're reusing loops and you're playing with loops and your creative act is not making a loop, but as you said, building a soundscape around some loops or some rhythms or some melodies and that's where you're focusing your attention, so you care less about the melodic content and it's actually nice to rely on something like Magenta to do that.

[SPONSOR MESSAGE]

[0:16:56.5] JM: Catch bugs before your users do with full stack error monitoring in analytics, for developers, by Rollbar. With Rollbar's error monitoring, you get the full stack trace, the context, and the user data to help you find and fix impactful errors superfast. You can integrate Rollbar into your existing workflow. You could send error alerts to Slack, or HipChat, or you can automatically crate new issues in Jira, Pivotal Tracker, or Trello.

Adding any of the Rollbar SDKs is as simple as a single command. Start tracking production errors in minutes. One cool feature is that GitHub repos could be deep linked directly to your stack traces in Rollbar. Go to rollbar.com/sedaily, sign up and get the bootstrap plan for free. Used by developers at companies like Heroku, Twilio, Kayak, Zendesk, Twitch, and more.

To check out Rollbar, go to rollbar.com/sedaily. Sign up and get the bootstrap plan for free. Thanks to Rollbar for being a sponsor of Software Engineering Daily.

[INTERVIEW CONTINUED]

[0:18:13.6] JM: The hooks in music seem to be only getting simpler. It's kind of strange. At least the pop music hooks only seemed to be getting simpler. I don't know where that ties in. Maybe you could say they're more elegant, and in that way, they're more complex. To me, they seem simple. They seem like something that a computer could understand and replicate quite well. It's the — Okay. We'll get into the surprise and the attention stuff, I guess, because that seems to be where the real difficulty lies.

We've been talking about generation, basically, the melodic generation. You've got this Google or AI Duet project where essentially you have a program that's just running and as you play the piano, the program is reading that dataset and learning and making melodies based off of that dataset. In one sense, that's incredible. In other sense, if you spend a lot of time in the deep learning space, it's not very surprising that we can have something that does that.

We had Feynman Liang on the show to discuss BachBot, which is a research for —

[0:19:26.6] DE: Cool! Awesome! I love his work. He's great.

[0:19:29.6] JM: Yeah, his work — It's incredible. He feeds in a bunch of Bach compositions and the machine learning models can create music that sounds like Bach. At this point, it doesn't surprise me that you're able to do that and you're able to do it on the fly. It's incredible, and I love it.

As you have discussed in the blog posts on the Magenta blog, in order to get really great music, you need the surprise and the dynamism that we hear — This is what makes pop music interesting. It's like you hear something, it's like the melody is not interesting or surprising, but it's got some weird — Maybe it's got a meme. It's got some weird pop memes that are going on, or some weird focal ticks, or some pitch bending stuff that it's almost like, "How would a computer be able to think of this?" How are you looking at the surprise and the dynamism area in a genitive fashion?

[0:20:29.4] JM: Maybe it's not surprising — Excuse the pun. That this ends up being extraordinarily hard. I think the thing I try to keep in my mind with respect to surprise in music is that it's always important to remember who's being surprised. By that, I mean an artist is generally making music for a specific audience. She may be focusing on teenagers who are coming to dance. She may be focusing on people who go to ballets, whatever. That artist is not trying to write music or make pictures or whatever art is. There's always an audience in mind.

I think when you talk about what's going on with pop music, you have to think about who the listener is. Behind that is, I think, some really nice neuroscience which suggests that our brains are rewarded very heavily by being surprised and then learning something from that surprise. Crucially, you need to be able to learn something from that surprise.

If you reflect on that idea, it actually yields kind of a nice formula for thinking about how to make great art. It's not an easy formula to implement, but the formula is understand your audience so well that you know what they know and then give them little puzzles, but give them puzzles that are just hard enough that they're fun to solve, but they can solve them. Don't give them puzzles that are so hard they can't solve them at all. That's not fun for them.

I think that's largely what's happening as an art form progresses. Consider something like punk music. At the time it was revolutionary and it seemed to break all the rules and it grabbed the young generation ready for something new. Then, we kind of learned what punk music was. I think, generationally, we sort of figured it out.

Now, actually, when I go back and listen to songs by The Clash, or Sex Pistols, they don't sound that revolutionary at all. I sort of learned all the surprise out of it. Also, note that at the time, what made that surprising wasn't really a huge shift in what came before. The Sex Pistols didn't break western tonality. They didn't stop doing things in four/four, they were doing songs in four/four that followed western tonal codes borrowed from African blues music, the blues, that which lead to rock. They played the same chords on the same instruments. They didn't even get new instruments.

They played them differently. They played them louder. They changed things rhythmically, lyrically, but it wasn't that huge of a change. I think that's the kind of surprise that is so hard to get computationally, because you're relying on knowledge of what the world knows. Your model needs to know what's normal. The model needs to know what's happening now, and then it needs to move away from that, but it can't move too far.

I don't want to go on forever. To close on that, the kind of surprise that's not artistic surprise is if you just get static on the line here or if the building collapses, or like that. It's really surprise with respect to what's normal.

One more example of that that I just been thinking about this week, which is — Even the most sort of bland pop music, great artists always managed to pull things away from boring. I've been thinking a lot about what happens — The advent of the click track. Music is being recorded against a metronome now, most music. You don't have this expressive timing and dynamics that I love where the band is speeding up and slowing down and you have that really nice violation of time.

I realized — This happened a couple of years ago. That by sitting on a click track, artists are able to be much dramatic about other kinds of timing changes they make. The example I had

was Adele. Honestly, I like Adele, I'm fine with Adele. I'm not putting on Adele in the car, but my daughter was really into Adele, my teenage daughter.

I was sitting at my DAW, at my workstation trying to key in the piano chords for one of Adele's songs so that my daughter could sing along with it. Kind of make some karaoke and we were going to record something and put it on YouTube.

Now, as I was listening with headphones, I was having a really hard time tracking the song and having a really hard time getting it right. I stopped and I started listening and I realized the reason I was having a hard time was that Adele was leading the beat and playing with the beat so much in her vocals that I was having a hard time following the underlying piano track. The piano track was locked in. I thought, "That's kind of cool." Even in pop music of the sort that Adele is doing, there's a lot of complexity happening rhythmically. The click track hasn't completely eliminated that.

[0:25:00.8] JM: It's totally in line with my thesis about where this stuff is going is. The more you factor out the work of the human, the more — It's like Parkinson's Law. The human just expands to take over creativity and take more creativity out of the smaller box. You put a smaller box around the human, the human will find more creativity within that space.

[0:25:27.8] DE: You told me that you have learned something new, I've learned a lot new in this interview. Parkinson's Law, I'm not familiar with this. Where did this come from?

[0:25:33.5] JM: Parkinson's Law, the idea is like the amount of work that you allocate to a task expands to fill the amount of work — The amount of time that you've allocated to the tasks. That's why you don't have an eight-week sprint, because people will take eight weeks to do that sprint. You have a two-week sprint, because people will — They'll get the work done in two weeks. It may be slightly substandard, but the density of work in two weeks will be higher than the density of work across eight weeks.

[0:26:04.0] DE: I didn't know the name of the law, but you work in software development long enough and you've certainly figured that out.

[0:26:10.4] JM: What you were saying about the surprise stuff — You could imagine a Magenta-like system that learns everything about pop music today and then extrapolates off of that and then has something that sounds — The thing about pop music is it's always s— It's borrowing 95% of the local maxima of the pop music that already exists. Then, it's got the 5% of surprise. If it gets the 5% of surprise right, then you've got a hit.

That's a pretty straightforward algorithm. You could imagine a Magenta-like system invoking that algorithm. That would have the sufficient level of surprise.

[0:26:56.0] DE: I think that's right. I think that's perfectly true. The other thing I would add is that if you expand that a little bit, I wish we could do a better job of having really focused datasets, because I think it's a little bit boring to train on all of pop music. It'd be much more interesting to have the last three years of a particular form of music that you like, or the last year of EDM and really be able to show the model, just that information and have it really pull signal from that and generate from there. Hack the data instead of hacking the model. Yeah, I agree.

[0:27:26.2] JM: Has Spotify expressed interest in working with Magenta?

[0:27:29.8] DE: Actually, that's great. No. We have had no talks with Spotify. Except, I can say that — Just personally, I worked on music recommendation for several years before starting Magenta. A lot of the guy that were in EcoNest that was acquired by Spotify and know a bunch of people in recommendation over there. No, they have not reached out and wanting to do some sort of weird licensing deal with Magenta. Anyway, it's open sourced. You're welcome to use it.

I do think — By the way. To the community, the low hanging fruit here is auto-generated music for your jog. The code is not there yet, but the idea that you might have your Fitbit and you're grabbing some sensor data and you're using that sensor data and you're using that sensor data to drive your job.

I'm surprised more people aren't playing around with those, because it would be fun and relatively easy to do just for yourself. You don't need to build this product that everybody wants to buy. I think as we move — Concretely, as we move TensoFlow to mobile in ways where we can do inference effectively and we get better, low energy consumption audio happening on

devices, this is like a real doable thing. I'd love to see someone do this and ping the Magenta list and we'll write a blog about it. We'll let you write a blog about it and we'll look into it and have fun with it.

[0:28:42.0] JM: That's definitely interesting. Speaking personally, what excites me about Magenta is what I said earlier about the melodic generation. Let's say I'm working in a digital work station today. I use FL Studio usually. Sometimes I use Ableton, but FL Studio mostly which has a midi interface. I could easily plug in some midi stuff. If I wanted to outsource my melody generation to Magenta, what would be the best workflow for doing that?

[0:29:07.5] DE: For FL, we don't have an instrument plugin right now. Right now what we have on Magenta open source on the GitHub is a workflow for using Ableton plus MaxMSP to route signals. It's a little bit clunky. We're working at — We're open-source, but there are no secrets. We're actually working to release a much cleaner Ableton workflow.

We'd be happy to work with anyone who's interested in helping build something similar. In the end, what you'll have — Right now, what you'll have is an instance of TensofFlor running on the same laptop and then that is already opening up a port and waiting to hear from you. Then, what we're is we're taking advantage of midi control signals to tell the model what to do. Fundamentally, what the instrument on the DAW side's job would be is to take your music and then shuttle that off to TensorFlow, but at the same time add the right control signals that say, "Okay. Here's the tempo and here's how many measures of music we want back," et cetera. That kind of thing.

It's pretty simple. It's not hard to do. The hard work has been done by us already, the API is there. It's just that without those control signals in the stream of midi, the TensorFlow model that's running the Magenta code won't know what to do. Does that make sense?

[0:30:21.3] JM: It does. What has been the reception of people who plug in to that workflow using Ableton?

[0:30:29.4] DE: It has been surprisingly positive, and I say surprisingly positive because I thought that I was the only type of person who had really had fun with this.

Right now, the first thing I would suggest is people should play with AI Duet. It's easy enough to find on the web if you search for Google Creative Lab AI Duet. That is running — Incidentally running on purpose a very, very primitive model. It's running our simplest LSTM model, fundamentally no different than what I did in 2002, trained on lots and lots of music. What it kind of does is noodles around.

We like that effect especially for novices, for people who aren't just musicians at all. It's really fun to play something and then you get this kind of weird noisy thing back that kind of sounds like what you played, but not much. It's very primitive, right? It's not something that we expect would carry a musician to want to play with it for a long time, because it's too primitive and you can't really drive it musically.

However, we already have up on our GitHub, we have the AI Duet code. Yotam, who wrote that code, graciously did a poll request against Magenta so that we could have a version of that code without having to do a fork. Now, we're repurposing that to be a frontend for playing around with Magenta models where you can change models, you can change parameters in the models and you can actually be a little bit more sophisticated in how you deal with it. Then, dropping in a more complicated model, a model trained on your own data, this starts to become really fun and it's a good direction to go in.

[0:31:58.8] JM: So much of music today is, as you said, about the loop, and you can imagine a workflow where I'm performing an AI Duet with the computer and recording three minutes of audio during that duet. Then, I can just play back that Duet and take small snippets of it that sound like a loop that I want to use, and I've got my hook.

[0:32:22.3] DE: We did a demo at a machine learning conference called NIPS, Neural Information Processing System and that was exactly the mode that we had. You could play for a while. If you like something, you could loop it. Catch it and loop it and it wouldn't change anymore, and we would leave that loop and it could continue playing in layer, which is a pretty nice workflow.

What I would say it like that kind of flow was really, really, really fun for trained musicians that came to play. People who weren't musically trained had a harder time with it. I guess that's not surprising. It's asking some sophistication. It's not too bad. We're okay if the tool is hard to use. All music instruments are hard to use.

[0:32:56.6] JM: Yeah. You could tell people, "Hey, just play white keys." That algorithm is not that hard. Many people, if they play white keys for an hour, they'll be able to make something that sounds reasonably catchy.

[0:33:11.3] DE: Jeff, you could make it easier and tell them to only play black keys.

[0:33:14.2] JM: That's true. That's even smaller. That's right.

[0:33:17.9] DE: You really can't lose then, right?

[0:33:19.7] JM: That's right. What about the other stuff? What about the instrumentation choice? Melodic composition is at least a well-formed problem, the instrumentation problem. Have you started to think about how to present the instrumentation problem, the soundscape problem, or the FX plugin problem, the mastering problem? Can you present these problems to an AI in a well-formed fashion yet?

[0:33:44.9] DE: I'm not sure. I think that there are companies that have tried to do automatic mastering and have had some success. One direction that we've put a lot of effort into is just generating new sounds, sounds that no one have heard before, no one has heard before, but that are musical.

We just released a paper and a dataset called NSynth. I don't know if you had a look at that. It's the most recent blog we have on the Magenta GitHub. What it is is a generative model that generates actually audio waveforms directly, like 16,000 times a second is generating the position of a speaker cone and we're sampling out new musical sounds and then building samplers from that, and that's really fun too.

[SPONSOR MESSAGE]

[0:34:31.5] JM: Do you want the flexibility of a non-relational, key-value store, together with the query capabilities of SQL? Take a look at c-treeACE by FairCom. C-treeACE is a non-relational key-value store that offers ACID transactions complemented by a full SQL engine. C-treeACE offers simultaneous access to the data through non-relational and relational APIs.

Company's use c-treeACE to process ACID transactions through non-relational APIs for extreme performance while using the SQL APIs to connect third part aps or query the data for reports or business intelligence. C-treeACE is platform and hardware-agnostic and it's capable of being embedded, deployed on premises, or in the cloud.

FairCom has been around for decades powering applications that most people use daily. Whether you are listening to NRP, shipping a package through UPS, paying for gas at the pump, or swiping your VISA card in Europe, FairCom is powering through your day.

Software Engineering Daily listeners can download an evaluation version of c-treeACE for free by going to softwareengineeringdaily.com/faircom.

Thanks to FairCom c-treeACE for being a new sponsor of Software Engineering Daily, and you can go to softwareengineeringdaily.com/faircom to check it out and support the show.

[INTERVIEW CONTINUED]

[0:36:11.9] JM: One of the projects I did look at was the RL Tuner which allows you to teach concepts of music theory to an LSTM trained to generate melodies. The idea here is that the long short-term memory machine learning model, it's already been trained to generate melodies. As we've said, definitely a plausible thing. It's probably no surprise to people who are familiar with this space, that you could build something that generates melodies.

Then, you teach it music theory concepts through reinforcement learning. You teach it a music theory concept, it generates a melody that uses that concept and then you reward it, because, "Hey, you successfully created a melody that uses that music theory concept." Is that the correct interpretation of that piece of research?

[0:37:02.4] DE: Yeah. Though, I would add — I think you nailed it. I would add that the fact that as music theory isn't all that important, what's important is that at least in terms of sort of machine learning ideas, models that — It's hard to build generative models. It's hard to build models that can generate new instances from trained on data. One; we're not sure why.

One reason is that models trained to reproduce data tend to play it safe. If they're being rewarded for — Let's say the job is to predict some Y from X, they're going to be rewarded to learn what the main modes are of that distribution. I don't know if this is getting too technical, but I'll go ahead anyway.

For example, it would generate blurry images, or generate boring images, or generate the equivalent of blurry music, which is sort of elevator music, boring music. Anything you can do to provide an extra task for that model is going to help with that problem. You could have a counterfeit detector that's trying to reward a model for generating a really genuine instance of what it's trained on. One instance of that is called GANS, very popular right now, Generative Adversarial Networks, or you could use reinforcement learning like we're doing and reward the model for also following some rules of counterpoint. Yeah, it works really well.

All we're doing with this work is following a general trend in machine learning this year to look at these issues, and I think we'll continue in that direction. If you want to summarize it for the non-machine learning people, the world needs a critique. Any artist needs a critique. What I would observe is that you can look at lots of photographs. You can look at millions of photographs and you could understand what photographs are, but it's completely different process to know how to take a photograph. It's not just a question of knowing the push the button and to focus the camera, but to actually actively be able to look at the world on a way that generates a good photograph is a different skill that perceiving photographs, and that's true for music as well.

What we're giving this model — It's really been trained to perceive music. It's going to predict what's coming next, kind of figure it out. We're saying, "If you want to compose, we're going to give you even more tutorial feedback. We're going to tell you, "Hey, wait a minute. That wasn't a very good composition, because you lacked this quality. Fix that." That kind of tutorial feedback ends up really useful for generative models.

[0:39:16.3] JM: You talked earlier about this idea of a product that would just make music for your run. You go on a run and you just start this program at the beginning of your run and it starts to generate music. If you don't like what you hear, you press X. You like what you're hearing, you press the green checkmark and maybe it accentuates stuff that recently changed if you pressed the red X it moves away from things that really changed or introduces something brand new.

I could imagine a very simple interface that's fun to use. It reminds me of the recent projects from Google around the drawings where you got this app where — Or it's like web app, I think, where it says like draw a house — You draw something and then the machine learning model tries to recognize what it is. You draw a little house and then it says, "Oh! That's a house." You're like, "Yes." It's this gamified thing that allows Google to learn and to develop a better product. The second iteration of that Google, the drawing product, has been this clipart thing that recently came out where you basically draw something and then it tries to generate clipart around that.

You see this really interesting workflow of Google products developing where your team gamifies the development of a product. Am I correctly articulating how you're thinking about bringing this sort of generative music to market?

[0:40:42.4] DE: Yeah. First, two things, the product was called — That experiment was called QuickDraw and it was done by our creative lab team. It's not meant to be a product. This was just kind of a way to highlight what we're doing with machine learning. It was a way to show off image recognition models. Frankly, I think it became a lot more popular than we expected. I guess that's happen when anything goes viral.

Sure, we've got — Basically, people are playing Pictionary along with the machine learning model, trying to communicate to the model; cart, horse, or dump truck, or a number of other categories. Recently, released another experiment where it will find the nearest piece of clipart to your drawing, that was last week.

This week, just yesterday in fact, we — Magenta group, the main person is David Hoff, just released a paper around generative models that actually draw new cats. It can see the pencil move, and we're going to keep moving in that direction. That was a Google research blog posting recently. Yesterday, in fact.

Yeah, I think gamify is one way to look at it, and I think that's the way we look — Definitely, the QuickDraw was gamifying it. I think more importantly, we need to listen to — We need to pay attention to what users are doing, and I think it's got to be done in a way that protects user's privacy, that's respectful to users and gives users what they want.

In terms of the jogging app, I think it's a great example. What I want from my job, it's not that I want to be musically engaged. I don't need — I'm really almost using — What I want is the signals in my ear buds to help me run faster. It almost requires that this model is listening and paying attention to me. Maybe it's paying attention to my pulse. Maybe it's paying attention to, like you said, a couple of buttons that I'm clicking as I go. In that case, I think most of the work is in paying attention and in learning from the jogger in real time what he or she needs. I think it's a great challenge. I don't know how to solve it quite yet, but it's a great challenge.

[0:42:38.4] JM: The incremental productization helps you get clues on how to solve it more generally.

[0:42:44.8] DE: Yeah, that's exactly right. I think we can't afford to think the way one might have thought 15 years ago, or even 5 years ago, that something like machine learning can happen in the lab and we'll get everything ready and then — We need to be doing kind of constant testing and trying things and we have to learn how to interact with people. I think HCI, human computer interaction, is going to be increasingly important area of research.

I'll give you a concrete example. Imagine self-driving cars. Waymo is working on self-driving cars, so is everyone else. One well-known problem, one well-known challenge for self-driving cars is how do you navigate with pedestrians? Don't hit the parent holding his daughter's hand to cross the street. Another problem is how do you communicate to that person that the car is not going to run you over?

We're used to making eye contact. Aren't we? You're at a stop sign, you're about to cross the crosswalk and you look over at the driver and you make eye contact, the driver looks back. It turns out that just figuring out a new social contract between pedestrians and self-driving cars is hard. What replaces eye contact? How do we learn to trust that these cars are safe?

That points to — In some sense, if you are to start to work with machines and make some sort of contract with that machine, that you're going to do something creative with it. How do you build trust with that algorithm? How do you build trust with Magenta to say, "Oh! We get it, right? I know what you're doing. You know what I'm doing. Let's work together on this." I think it's a really interesting challenge and one that if we can make some strides, we'll build much, much better tools for creative people.

[0:44:26.4] JM: All right, Doug. It's been a real pleasure talking to you and I'm sure we'll find a reason to do another show in the near future. I'm a big fan of your work and I look forward to using it in my music composition.

[0:44:39.0] DE: Yeah. I didn't know before getting a chance to chat with you, Jeff, how much of a musician you are. Where do we find your music? Can't we do a shout out for you here before we close?

[0:44:48.2] JM: Sure. It's on Spotify, if you look up the_prion, or Soundcloud. Soundcloud the_prion is also workable.

[0:44:59.5] DE: Isn't prion — Isn't that mad cow disease? Isn't that what it —

[0:45:02.5] JM: Yes. Mad cow disease is caused by prions, but the prion protein phenomenon itself is as interesting as it is horrifying.

[0:45:13.8] DE: Promise me you won't edit the prion part of this out.

[0:45:17.7] JM: Okay. I won't edit that out.

[0:45:18.9] DE: Okay. It was a great conversation.

[0:45:20.0] JM: Talk to you soon, Doug.

[0:45:21.0] DE: Okay.

[0:45:21.3] JM: Yeah, great conversation.

[0:45:22.3] DE: Take care.

[0:45:22.7] JM: Okay. Later.

[END OF INTERVIEW]

[0:45:27.5] JM: Thanks to Symphono for sponsoring Software Engineering Daily. Symphono is a custom engineering shop where senior engineers tackle big tech challenges while learning from each other. Check it out at symphono.com/sedaily. That's symphono.com/sedaily.

Thanks again Symphono for being a sponsor of Software Engineering Daily for almost a year now. Your continued support allows us to deliver this content to the listeners on a regular basis.

[END]