EPISODE 1118

[INTRODUCTION]

[00:00:00] JM: Chatbots are useful for developing well-defined applications such as first contact customer support, sales and troubleshooting. But the potential for chatbots is so much greater. Over the last 5 years there have been numerous platforms that have risen to allow for better, more streamlined chatbot creation.

Dialogue software enables the creation of sophisticated chatbots. ParlAI is a dialogue platform built inside of Facebook. It allows for the development of dialogue models within Facebook. These chatbots can remember information from session to session and continually learn from user input. Stephen Roller is an engineer who helped build ParlAI and he joins the show to discuss the history of chatbot applications and what the Facebook team is trying to accomplish with the development of ParlAI.

We are looking for writers and researchers to help with Software Engineering Daily. If you are interested, send me an email, jeff@softwareengineeringdaily.com. I'm also looking for investments, infrastructure companies to invest in. If you're building an infrastructure or developer tooling company, send me an email, jeff@softwareengineeringdaily.com.

[SPONSOR MESSAGE]

[00:01:13] JM: When the New Yorker magazine asked Mark Zuckerberg how he gets his news. He said the one news source he definitively follows is Techmeme. For more than two years and nearly 700 episodes, the Techmeme Ride Home Podcast has been one of Silicon Valley's favorites. The Techmeme Ride Home Podcast is daily. It's only 15 to 20 minutes long and it's every day. By 5 PM Eastern, it has all the latest tech news, but it's more than just headlines. You could get a robot to read you the headlines.

The Techmeme Ride Home Podcast is all about the context around the latest news of the day. It's top stories, the top posts and tweets and conversations about those stories as well as behind-the-scenes analysis. Techmeme Ride Home is like TL DR as a service. The folks at

Techmeme are online all day reading everything so they can catch you up. Search your podcast player today for Ride Home and subscribe to the Techmeme Ride Home Podcast.

[INTERVIEW]

[00:02:18] JM: Stephen Roller, welcome to the show.

[00:02:20] SR: Thanks for having me.

[00:02:22] JM: You work on dialogue research at Facebook. Dialogue research, I think of as a better way of describing chatbots. So I may refer to chatbots and dialogue research interchangeably during this episode. Why are chatbots useful?

[00:02:41] SR: I think it's a longstanding dream in the field of artificial intelligence and computer science, right? We've always had the dream of what if I could just talk to my computer like I talk to other people and converse with them and they would know what are my intentions? Why am I asking them this? What can they do to make my life better or help me along? And programming is fun. I love programming, but it'd be nice to just communicate with computers as well.

[00:03:14] JM: And what about your personally? Why are you personally interested in chatbots? Why have you pursued this line of work of all the different kinds of computer science research you could be doing?

[00:03:25] SR: Yeah. I first stumbled into this area. I joined FAIR after my PhD where I did work in natural language processing. But my work, my background work was like a little bit more linguistically oriented. Focused on like the meanings of words. A little bit more like what can computers teach us about language rather than how do we teach computers to understand language?

So when I joined FAIR, I was looking for projects to join, projects to collaborate on. So I gave Dialogue a shot, and it turns out to be like a really great lend for me. A mix of novel research using the greatest and latest neural networking and machine learning techniques as well as a

bunch of interesting software engineering problems and opportunities for scaling and things like this.

[00:04:15] JM: What are the domains in which conversational agents or chatbots are actually useful today. What are the places where they're useful versus the places where they're not so useful?

[00:04:28] SR: Meaning like what's the state of the art?

[00:04:30] JM: What are the domains we've actually conquered rather than the ones where we're stumbling?

[00:04:34] SR: Yeah. I think, obviously, you've seen a big wave of digital assistance. And I think that work is very exciting. A lot of what we focus in ParlAI and the dialogue research at Facebook is on open domain chatbots, which are chatbots that can talk to you about like literally anything. And they're usually not focused on accomplishing a task for you or rather like the goal is just to have a conversation with you about anything for as long as possible.

That's primarily where we work. And I think we've seen a lot of exciting advances over the past couple of years to where I think if you were to try some of these newer chatbots that have come out in the past year, you'd probably be really impressed with like, "Oh wow! This was further than I expect." I think what you see commercially is a lot of digital assistance, a lot of success in this. A lot of customer service type things. But what I'm really excited about when I think where things have made big strides in the past year is on these open domain chatbots that can actually talk to you about anything.

[00:05:47] JM: Really? As in I can sit down with a chatbot today and say, "Hey, what's the weather? And what is this spot on my skin? And recommend me a restaurant that looks appealing." I can ask all of these things of a chatbot today.

[00:06:05] SR: Yeah. I would put those still on the category of like task-oriented tech chatbots, where there's like some goal on mind, like answer my medical questions or tell me about the weather. What I really tend to work on and what I'm excited about is these social chatbots,

right? Rather instead you might ask what's your favorite chess move? And it might go into detail about like what its favorite move and why that is? Or what's your opinion of the fall of the Roman Empire? Let's debate that? So it's less about what can you do for me and more about like let's enjoy the experience together.

[00:06:45] JM: So can you give a few more descriptions of what a general purpose chatbot would be doing?

[00:06:54] SR: Yeah. I think the endgame here or rather like where a true general purpose chatbot should do both of these things, right? If I ask it to book me a calendar invite, then it should absolutely help me out with that. Where I focus my research is the sort of social part. And so I think we should see a mix of these.

The bots that we've been developing really focus on having a few different behaviors that I don't think you tend to see in this sort of assistant type chatbots. They have persistent personas. So they'll have like personalities like I love basketball or I have friends in the tech industry or something like this. And they can use this information like consistently refer to it.

One of the other attributes that chatbots that we're working on developing and believe, what's really important is empathy? A chatbot should understand as its talking with you what's your mental state? How are you feeling? And respond emotionally appropriately to that. If you say something like my dog just ran away. The chatbot should respond appropriately with something like, "Oh, I'm really sorry to hear that." But if you say something like I just got a promotion, then the chatbot should respond, "Oh, that's great to hear. Congratulations." So responding emotionally appropriately is a really important characteristic of an open domain chatbot.

The last one is general knowledge. So I think when people talk to chatbots, they expect it to know things about the world. Some of that is encyclopedia, some of it is commonsense reasoning. But if I ask what's the tallest building in America, I sort of expect the bot to be able to have this information available to it and be able to answer some questions like that. Or even integrate this information in common dialogue. Just, "Oh, by the way. What's interesting," or a funny story about that sort of things. So consistent personality, empathy and a knowledge about the world. These are all things that I think are really important in the general purpose chatbot.

[00:09:07] JM: Empathy. You mentioned empathy. That's one of these things that requires some maintenance of context as to what is going on in the conversation. The chatbot needs to be able to acknowledge the perhaps sad state of happy state of the person its interlocuting with. Tell me about context. How does a chatbot retain and understand context in a conversation?

[00:09:37] SR: Yeah. There are a few different ways that people go about it. So one of the more classic ways that people will do and approaches that people will take is they'll do something old like dialogue state tracking. So you'll have like some information about the dialogue like, "Oh, the user is asking about a restaurant and they wanted to be on 5th street." And you'll have this as some sort of state that you could do, say like some sort of database query about and help them.

Then like when you're doing dialogue research or building a chatbot, the task becomes a lot of like keeping track of that state and updating that state as the conversation goes on. The way that we often approach it is in a much more like raw neural network fashion. So we just like input all the dialogue context, all history as one big string. Sort of saying person one said this. Person two said that. Person one said this. Person two said that. What do you say next?

So we just treat these things as raw strings and have it input that and the model has to figure out, "Oh, what do we do with that? How do I respond? Who said what?" All of these has to be learned from scratch.

[00:10:51] JM: I'd like to get into a little bit more of a conversation about Facebook. If you imagine Facebook in 5 years, what are the tasks that you envision dialogue models fulfilling for Facebook?

[00:11:07] SR: Oh, that's an excellent question. I think there are all sorts of places where dialogue models can be helpful to our users. We already have this product called portal, which is a really excellent product that lets you make video calls with people. One of the things you can do is you can already say, "Hey, portal, call mom." and it stats dialing up mom. And that's super nice. Some of the other things might be as I might have an assistant on Messenger that's helping me keep track of what's going on with my friends. I might say, "Hey, assistant, what are

the latest updates of my friends?" and it could integrate that information and look through my newsfeed for me and say, "Oh, hey! Your friend Jim has a new photo. He got married." Oh! That'd be really cool, right? I think there are a lot of places for dialogue to be part of Facebook products.

[00:12:06] JM: You work on ParlAI, which is spelled P-A-R-L-A-I. What is ParlAI?

[00:12:14] SR: ParlAI is a platform for doing dialogue research. It's an open source platform. It's gotten over a hundred contributors and it's got everything you would need to do to do dialogue research. So whether that's I want to collect a new dataset or I want to train a new model, and there are all these existing datasets out there and I just want to use those without trying anything. Or I want to create a new model and I need to compare it to baselines, compare to other approaches that people have tried before. I can just sort of try those different models. It's got a model Zoo, "Oh! What are the pre-trained models that somebody else has released and can I leverage those when building my new chatbot?" And then it's got everything you would need to also evaluate a chatbot.

So once you have a chatbot, unlike a lot of areas in AI research, it's not always clear how to evaluate it, except to have people talk to it. So we've got all the tools that you need to connect your chatbot up to Amazon Mechanical Turk and have people chat with it and give evaluations or give performance ratings, things like this.

[00:13:20] JM: Go a little bit deeper into the problems that ParlAI solves for researchers that are using it.

[00:13:26] SR: Yeah. I think there're quite a few different problems especially in all those different spaces. When you're a researchers and you want to create a new dataset, that let's say you want to create a new dataset that teaches the model how to have empathy. The thing you're going to do is have humans talk and exhibit empathy and annotate their utterances with this sort of empathetic information. So we have tools so that you can like quickly spin up a user interface where you can have that chat where you can annotate that information and sort of build what you need for that.

So similarly with evaluation, if I need to connect to Amazon Mechanical Turk and have people talk and evaluate. I don't want to have to spend so much time focused on building the UI of this tool or dealing with the engineering of connecting with Amazon Mechanical Turk and pairing humans together. So we abstract a lot of that away from you.

One way we do that is by treating all things in the world as agents. Whether you're a dataset or a human talking to it on Mechanical Turk or human talking to it at your local keyboard, or an AI agent, everything is an agent. So we treat them all the same. This gives us really nice abstractions to work with so that we can sort of plugin the AI model in place of a human or plug a human in place of a model really easily.

[00:14:55] JM: ParlAI makes available a wide set of datasets through its API. How does ParlAI use these different datasets?

[00:15:04] SR: Yeah. We have over 100 datasets in ParlAI. Some more from our group. Some of them were from external groups. And so one of the things we really focus on as a first-class feature in ParlAI is if I want to train an AI agent that exhibits multiple behaviors. I'm going to train on multiple datasets at the same time.

It's really easy to just sort of say, "Okay, give me dataset A, dataset B, dataset C," and start training all three of them at the same time and get a model that can do all three of these behaviors. So that's sort of first-class functionality within ParlAI.

When we were talking earlier about having a model that exhibited this sort of consistent personality, empathy and knowledge, we did this with a sort of multicast training as it's called, where we train on all different datasets. If you're a new user or a new researcher who wants to come into dialogue, you can sort of take the stock of what are the datasets out there are ready? And just start utilizing them as needed. And if you want to mix-and-match behaviors, no problem.

[00:16:10] JM: Described the workflow for training a chatbot with ParlAI.

[00:16:14] SR: Yeah. We have a lot of – It's a very command-line-heavy utility or command-line-heavy platform. So if you want to train a new model, it can be as simple as calling the train

model command from the command-line and then you just sort of say, “All right. Here are the tasks I want. Here's the model, the sort of model that I want. What’s the model architecture?” and things like this. And here's the learning all the other neural network parameters. And you hit go and it starts training.

A lot of researchers want to do more sophisticated things. Maybe make a custom architecture or make a custom dataset. And it’s really easy to just sort of build only the part of the dataset that you need or build only what's special about your model. So you might go and write a little bit of custom code utilizing our sort of abstract base classes and things like this. And you’ll be off on your way training your special model. And if you don't want to mess with data, you don't have to mess with data. You can just use all the existing datasets. If you don't want to mess with modeling, you don't have to mess with modeling. You can just adjust the data and start training existing models already.

[SPONSOR MESSAGE]

[00:17:29] JM: Today’s show is sponsored by Datadog, a monitoring and analytics platform that integrates with more than 250 technologies including AWS, Kubernetes and Lambda. Datadog unites metrics, traces and logs in one platform so that you can get full visibility into your infrastructure and your application.

Check out new features like Trace Search and analytics for rapid insights into high-cardinality data; and Watchdog, an auto detection engine that alerts you to performance anomalies across your applications. Datadog makes it easy for teams to monitor every layer of their stack in one place. But don't take our word for it, you can start a free trial today and Datadog will send you a T-shirt for free at softwareengineeringdaily.com/datadog. To get that T-shirt and your free Datadog trial, go to softwareengineeringdaily.com/datadog.

[INTERVIEW CONTINUED]

[00:18:32] JM: With ParlAI, you are training these chatbots, and you need to be able to have some understanding of what is a successful session and what is a failed session. If I am engaging in dialogue with a chatbot, I need to somehow give feedback to the system whether

my conversation has been a successful one or a failure. How does that labeling workflow proceed in ParlAI?

[00:19:01] SR: There are a few different ways to go about it. As I said earlier, we tend to focus on open domain chatbots. And one of the things about open domain chatbots is it can be hard to define what is a successful interaction, right? As long as we talk and we enjoy talking together, that can be viewed as a success. And so the valuation in this sort of open domain chatbots can become quite difficult.

To that end, we have a number of tools available to implement like existing architectures in the literature, different evaluation methods in the literature. One of the more classic ones is you might just have somebody chat with your model for a few minutes, maybe 10 utterances. And then you ask them, “Hey, on a 1 to 5 scale, how much did you like talking to your partner here?” And then you would compare that against your previous version of your chatbot. Or maybe even compared to a human, how far away are you from human performance? And if you wanted to do that, you don't have to write any code whatsoever.

One of the need things that we've developed in the past year is this thing called acute ACUTE-Eval, which is actually even more efficient way of doing this. The problem when you do these like sort of classical Likert scale, on a scale 1 to 5, what’s your opinion here? Is that if I don't evaluate two different models at the exact same time, well then I’m having a different pool of people give their opinions. And so if I am asking on a Tuesday night versus a Saturday morning, there might be different people working and it might give systematically different ratings.

So one thing we've developed is this tool called ACUTE-Eval, which actually shows people pairs of conversations with the AI's highlighted different colors. So I’ll show a conversation between human and a model on one side, and the model will be blue. And then on the other side of the screen, I’ll show a conversation between a human and a different model, and that model will be red. And I’ll say, “Hey, which of these two speakers would you rather talk to for a long conversation? Red one or blue one?” And you get their opinion. If you do this enough, then you can actually get a pretty nice estimate of which model is better than the other. And this allows you to do like really nice fair comparisons with other people in the literature without having to access their model. I only need access to conversations they've had with people, right? And the

other nice thing is that it's quite a bit less expensive. It's less labor-intensive to do this, because I don't need to have you sit there for 10 minutes and actually talk to the model. I just need to have you sit there and rate the model. Is it better or worse than this other conversation?

What's neat though is this technique works really well in self-check mode. So originally we tried, "Okay, let's have a model talk to a human and then we'll have people rate that against another model." What we actually found works really well is we'll have a chatbot talk to itself and then we'll compare that to another chatbot talking to itself. And if you do this enough times and get enough annotators and random pairs, when you can get a really strong estimate of the same performance, we found that it correlates really well with the traditional ways of doing human evaluation while being like 10X cheaper. So it turns out then, you can sort of get a human evaluation of your model in like 30 minutes. Works really well.

[00:22:37] JM: So you're describing there a kind of adversarial situation where you can have two models that are playing against one another and your you're able to essentially fulfill the role of the human interlocutor with an agent, if I understand correctly.

[00:22:57] SR: Yes. What you do is you have – You'd have a model and you like seed it with a little bit of information or just a random seed, random.seed, right? And start talking. And then you have another model that reads what its partner said and responds, right? And you just have them go back and forth like 10 turns. And you can collect infinite of these. You can just start having your computer crank out a bunch of these conversations.

Historically, the models I don't think were very good. They weren't very high-performing. But in the past year or so, we've seen this real leap forward in terms of how high-quality these models can be and they're actually able to carry a conversation. And so what you find is that when a model does really, really poorly, it will like go off the rails. Maybe start repeating itself. It might get stuck in a loop or just start ignoring its partner. But a good model won't do that, and you can really see this behavior in the self-chat mode.

[00:24:03] JM: So let's talk a little bit more about the role of a labeling. Tell me about how Mechanical Turk plays into the development of chat models.

[00:24:16] SR: Yeah. With these neural network architectures that you train in the sort of end-to-end raw conversation and predict the next utterance, having examples of people displaying the information that you need is really important.

For instance, when we wanted to teach this model to be knowledgeable, we set up – It's called a Wizard of Oz type experiment where you have one person sit-in and pretend to be the AI that you want trained. So you'll give them a user interface. And so for this knowledge one, which we call the Wizard of Wikipedia, we gave these Mechanical Turk users access to a search engine over Wikipedia. And so while they were having a conversation with the person, we would be searching Wikipedia, "Hey, here are some information that might be relevant to this conversation. If you're going to use this information, will you select that information and click, "I'm going to use this sentence from Wikipedia and integrate it into my reply." And then write your reply.

So now what we have is a whole bunch of examples of humans talking to other humans, integrating knowledge into the conversation, and we have a grounding of what it is that they were reading while they were answering this other person so that we have examples of how to read Wikipedia while you're talking to somebody and answer their questions intelligently.

That's just one example. We'll do all sorts of things like this. Like for a persona chat for this consistent personality, we had people assign them these sort of random personalities. Mask them to role-play as this personality. So you'll have, say, I'm a basketball lover from Michigan and my favorite artist is Prince, right? And then say, "All right, I need you to role-play somebody who is like that." And then what we'd get is a dataset of people having conversations about themselves while having this sort of background information about them.

So when we can sort of build these UIs really quickly to collect these annotations, collect these examples of behavior, it becomes quite interesting and quite easy in order to just integrate this information into the chatbot while it's talking with you.

[00:26:40] JM: There is a term, continual learning, that I'd like you to explain. Could you explain what continual learning means?

[00:26:47] SR: Yeah. Continual learning is the idea that a model should be always learning or an agent should be always learning as it continues to talk to people. So the way a lot of AIs, a lot of machine learning works these days is I'll have a dataset on disk and I'll go and I'll have my GPUs process that data and process the neural network and they'll learn all these data. And at the end I'll end up with a model that can do whatever behavior that I trained it to do. Awesome.

Now when I go to deploy a chatbot, I want this model to be continuously learning from its experiences. That dataset is old. It's stale. It was collected at some point in the past. As new things happen, you want the model to learn and responds to that. When this episode comes out and we say, "Oh! There is a new interview about ParlAI, you don't want to have to go and train the model from scratch in order to see and understand how people are conversing about this new piece of software, ParlAI. Rather, instead you want them to be as you chat with them, integrating this new information, adding into their knowledge base, learning by example from talking just like a human dose.

[00:28:07] JM: Tell me some of the important traits of a successful chatbot. What should chatbot be able to do? Let's just re-level set the conversation around the basics of a chatbot.

[00:28:22] SR: Well, at the core level, a chatbot should be able to talk with you. Whether that's your messaging it on instant messenger of some sort, or whether you're maybe talking to it and it has speech recognition and things like this. You should be able to talk to the model. It should be able to understand what you're saying and respond appropriately. Perhaps performing some sort of action for you like booking a calendar entry while it's doing this. But you expect it to talk back with you, right? It should respond.

[00:28:54] JM: And there is a distinction we can draw between the dialogue systems of the past that were structured and had this task orientation. There's a newer way of doing things that is a deeply trained end-to-end system. How do these two types of training models compare on the requirements needed for a chatbot?

[00:29:24] SR: Yeah. I mean, I think a lot of it is around where the advances of calm and AI and NLP and natural language processing over the past couple years. What we're really seeing is leaps and bounds in this sort of end-to-end networks. When I say we're focused on an open

domain chatbot, what I really mean is I want to chatbot that can talk about these things and I'm going to train that in an end-to-end fashion. But that's not to say that you couldn't say train a task-oriented model in this sort of fashion as well.

I think what's interesting is that this sort of traditional way of doing task-oriented dialogue where you have some sort of dialogue state that you're keeping track of and you're trying to explicitly model in symbols. What is this other person thinking and what is it that they want from me? Is that you have a really hard time with these sort of open-ended requests or request that you haven't seen before.

So if I haven't ever seen somebody talk about that symbol, maybe I'm going to have trouble producing that symbol or even thinking out of the blue like what's that symbol? When you do these end-to-end models, you're really treating these things as continuous representations? You're removing the aspect of symbols here and you're just saying, "Okay, I want you to think about this and process it. It learns what is the relationship between English words. How do I construct this sentence? How do I respond appropriately? And it learns these things.

So what you're really seeing is a difference from, "Okay, let's update and keep track of symbols to let's let it all be hidden in the information of the model. So let's let it learn from scratch what it needs to do."

[00:31:19] JM: And is there any difficulty in maintaining the deeply trained models. It sounds like it will be harder to understand what is actually going on in the training process and how those models are coming to fruition. Is it hard to maintain a deeply trained end-to-end system rather than a highly structured system?

[00:31:43] SR: Yeah, that's a really good question. I think it requires a different way of thinking about it. Rather than thinking about, "Okay, at this point in time, this is what the state looked like." And so I can reason on what's the next sort of states that it will go to? That's sort of the traditional way of thinking about it. And I think when you have to start to maintain these end-to-end models, we have to start to think about, "Okay, what are the aspects of my training data? What is the model seeing people do before?" Because it's going to learn to exhibit bad behavior. So if you start seeing it misbehave and say – Guess that somebody wanted a book instead of a

movie. Or respond about the wrong topic. You have to start to look, “Okay, where are my training data? Were there mistakes like this where somebody confused a book for a movie? Or somebody changed topics rapidly? And you have to start to think, “Okay, how can we fix the data? Or how can we adjust the data? Or how can we augment the data in order to get around these issues with it?

You also I think exhibit – These models exhibit failure cases, like they often tend to repeat themselves and get stuck in. And so then you start to say, “Okay. Well, what is it about my architecture? What is it about my information that's causing this thing to get stuck on a loop?” And you start looking for, “Okay, how do I teach it not to do this?”

So it becomes a different sort of debugging. Rather than sort of setting a breakpoint and saying, “Okay. Well, what happens next as it transitions from this state to that state?” You have to start to say, “Okay, where am I seeing these issues in my data? Or why is my architecture have this sort of repetition bias or something like this?” And you start solving the problems in that way.

[00:33:35] JM: I’d like to talk more about how to use models that are built with parlay. So there are some pre-trained models that come with ParlAI? Can you explain what a pre-trained model is and an example of how you would use that?

[00:33:52] SR: Yeah. Pre-training is this idea that I'm going to take this neural network. Perhaps it’s very large and perhaps it's very expensive to train. But I'm going to train it on all these data ahead of time and I’m going to do that exactly once. Then I'll have these sort of pre-trained weights initialized and I can then fine-tune the model to exhibit very specialized behaviors. So I can say start with a very generic chatbot that has just seen a lot of texts of people chatting about pretty much anything. And then if I want to teach empathy, then I'll spend just a little bit of time showing the model examples of how to be empathetic. Or if I want to train this model to book calendars, I’ll show it just a few examples. Maybe a couple hundred or a thousand, maybe more if you have them, but I'll just show those few hundred examples of that behavior and I'll spend very little time both in compute and then in wall clock time. Actually teaching it this specific behavior.

The thing about pre-training is it can be very expensive to do the pre-training, but very, very cheap to do the fine tuning. So you may only need access to a GPU for eight hours or something to sort of specialize the model for this behavior. So we released a bunch of pre-trained models that we've trained that have this sort of general properties that have seen a lot of texts. And this makes it really easy to sort of start and focus on just the behavior that you want to exhibit. And you don't have to spend nearly as many computational resources to obtain that model. And we've released dozens of pre-trained models from different research papers or ones that are really good starting points for all different types of tasks.

[00:35:52] JM: ParlAI also contains reference models. What is a reference model?

[00:35:57] SR: So what we mean by reference model is models that are likely to be compared against in a scientific setting. So if I have a great research idea and I want to adjust the model and say, “Okay, I’m going to give access to this extra information. Or I’m going to add this extra little neural net component over here.” I’m going to need to compare against the baseline right? So that baseline is usually going to be something standard in the literature or it’s going to be something that lots of people have tried before and they know how it behaves. Or it’s going to be something that's just like simple and very closely aligned. So we have implemented quite a few reference models of existing things that comment the common implementations that you’ll want to compare against if you’re developing a new research chatbot.

For a lot of people, they don't necessarily do want to focus on a research chatbot. And so they might also just want to use these models off-the-shelf. So if you’re just sort of looking to say, “Okay, give me one of these awesome neural nets that exist today.” You don't have to implement it any of that yourself. We've implemented that already and you can just sort of import that and refer to it and use it.

[00:37:12] JM: Is it useful to build multiple models or to use multiple models and compose them together into some kind of higher level architecture?

[00:37:21] SR: Absolutely. Absolutely. Earlier, I was talking about how we have a model that can read Wikipedia while it's talking with you. The way this is actually implemented is we have a model that searches Wikipedia and finds relevant information for the current conversation. And

then we have another model that learns to ingest whatever information was selected by that first level. So in that way you can compose these two models and sort of get this higher level behavior of reading Wikipedia by breaking it down into a searcher and then just an ingester.

So we have a lot of instances of compose models. The knowledge one there is one example. But another one is, say, you might want a safety classifier that's going to make sure that your model isn't going to say something offensive to users. You might want an extra backstop protecting your users. You might want to compose these models. So first you'll have your neural network that generates a response and then you'll have a safety model on top of that that acts as a backstop and says, "Oh, actually I don't think we want to say that here." So we have a lot of different ways of composing different models. Yeah, you can mix-and-match however your needs are.

[SPONSOR MESSAGE]

[00:38:49] JM: Operations teams can find themselves choosing between two options, either take full control over the infrastructure yourself or give all developers permission to access production. The first approach increases the operations team's workload, which often results in overwhelming situations and ops becoming a bottleneck. And the second approach allows for rapid deployment of changes to production, but it causes a serious risk to infrastructure uptime.

With Octopus Deploy, operations teams have a third option. The Octopus platform can be authorized to run approved steps and play an intermediary role. Octopus delivers self-service without sacrificing control over production, and it also provides a comprehensive audit log of the changes that are being made. Developers can enable self-service with automation. By automating the processes that are forming a bottleneck, developers can free themselves from the waiting game.

You can check out the most frequently requested run book templates by going to octopus.com/selfservice. You can find those run book templates at octopus.com/selfservice. And I want to thank Octopus for being a sponsor of Software Engineering Daily.

[INTERVIEW CONTINUED]

[00:40:04] JM: As you mentioned before, ParlAI has been used for an experiment in making more empathetic chatbots. Can we talk about that example in more detail? How did you design a more empathetic chatbot using ParlAI?

[00:40:21] SR: Sure. That was work done by my colleagues and the internment that we had a couple summers ago. [inaudible 00:40:26] was the research scientists leading that project. The way they did it was they had people have conversations with each other and they asked them to annotate and say, "What do you think is the right emotional state to respond with?" And then there were other cases. So they did that for a little bit and then they had cases where, "Okay, I want you to jump in this conversation and I want you to respond in this particular way. So I want you to the respond with anger. Or I want you to respond with sadness. Or I want you to respond with happiness."

And then once you have examples of that behavior, you just sort of teach the model to predict what's the right emotion here. And then once you have that sort of prediction of the emotion, you can teach it to generate a response with that specific emotion. So then you can sort of control the model in a few different ways. So you can sort of say, "Okay, now I never want you to respond angrily." That's not behavior I think my chatbot should do. If the model wants to respond angrily when I say, "No. No. Instead, I want you to respond happily, or sarcastically, or something like this."

So collected a bunch of these conversations displaying these emotions and then train models to exhibit these emotions and then ask humans, "Hey, which are these models do you enjoy talking to more? And what should these models do you think displays more empathy?" So the models that were trained have these sympathetic responses. Exhibited more empathy and were more entertaining to talk to.

[00:42:05] JM: You've mentioned this close integration with Mechanical Turk. Could we go a little bit deeper into that? Describe what the integration with Mechanical Turk gives you access to.

[00:42:16] SR: Yeah. A lot of people use Mechanical Turk for these sort of static tasks. I want you to find me the stop sign in this image. So then they'll upload this sort of static CSV file or whatever. Here's an image. I want you to find the stop sign. And the user will would just load up that example and draw a box around the stop sign and then return, submit the work for a review. This is really great. Mechanical Turk makes it really easy to do these sorts of annotations. Where things get really tricky is when things are dynamic.

The thing about chatbots is I can't just evaluate by having a static conversation. Every time we converse, it's going to be slightly different, right? So we've really build tools for plugging in these models or collecting these conversations in a non-static setting where something is going to be responding to the user and it needs to integrate that information and then return it back to the Mechanical Turk user and show this next chat message or show this information to them. So that's what we've really made fundamentally easy.

The other tricky thing about Mechanical Turk is sort of expects, "Oh, we've just got one user, and they're assigned to this task." But with chatbots, what we really usually want, very often want, is two humans talking together. That's how we collect these datasets. So we've made it really easy to pair people together or even groups of people together so that you can form a sort of chat room with multiple humans in it. And this can this can all happen on the backend of ParlAI and you don't have to worry about the details of queuing people up to be matched together, things like this.

[00:44:10] JM: Give me more of a description for the software architecture around training and deploying a model that will be used with ParlAI. What are bounds? Like the bounds around ParlAI and where does it get more into the machine learning frameworks like Tensorflow or PyTorch?

[00:44:29] SR: Yeah. At the end of the day, a lot of these agents will boil down to a bunch of PyTorch code running. What ParlAI really makes it easy to do is it facilitates getting the data into that model, formatting it in the exact right way. Doing sort of information search around that response. Just because you have a neural network doesn't mean that it's going to respond correctly. So maybe you need to do some augmentation. How do you turn that output of that neural network back into text that a human can receive a reply?

All of that is things that we've extracted into this notion of an agent. And so when you're working on a neural net, all you have to do is focus on the neural net part and all that information of like bringing it together, formatting it into the into the way that the neural network expects as input. Turning that output back in the text. This is all sort of abstracted away.

One of those really interesting things is that most machine learning problems are sort of static. So say I'm doing machine translation. I have a French sentence in and I have an English sentence out. And so every example is independent of all the other examples. With Dialogue, there's actually a multi-turn conversation part of it. So one of the tricky things is in order to fully utilize GPUs, you need to batching and group up multiple examples that the GPU can see it all the same time. And this turned out to be really tricky to do in Dialogue, because different conversations are different lengths. So you might be finished up with one conversation and just beginning another conversation. But will it automatically do that batching for you? So all you have to do is really handle examples.

And then once you have a neural network, you want to deploy it. Now, because we have these abstractions of text in, text out, we can sort of treat your model as a black box and use it, deploy it, say, to Mechanical Turk or to a chat service of some sort, like Messenger, and just input the text to the model and handle and it produces a response and will handle delegating, "Oh! Okay, this conversation, we're talking to this user. And so you need to – Here's your conversation history. But over here, at the same time, you're having another conversation and we'll keep track of that. So make sure that your model has only seen the conversations that it's replying to right now.

[00:47:14] JM: How have companies outside of Facebook used ParlAI?

[00:47:19] SR: So we've had a lot of contributors and a lot of participants. I think we've had over 5,000, 6,000 forks of ParlAI at some point. Maybe only a thousand. But we've had a lot of different people take ParlAI and extended it for whatever their needs are. A lot of those people are often university researchers. Sometimes those people are different companies. One thing that stands out to me is we held a contest a few years ago for this conversational AI contest where we were first – This is back in the early days when we were just experimenting these

consistent personalities. We had a contest where you could develop different chatbots in ParlAI and we would have them all ranked in a giant competition, say, “All right, who can build the most entertaining chabot with this data?”

And Hugging Face was one of the top participants of that competition, and this is back in the day when they were focused on building chatbots. So at the time, one of the things they were doing was using ParlAI to build dialogue systems. That was a couple years ago now. Since then, I have seen other companies use ParlAI in ways I didn't expect. Since we released this model Blender bot, we've gotten a lot of interest in people looking to use our models and use our framework in order to build specialized chatbots.

[00:48:49] JM: What are the goals with the future of ParlAI? What are the gaps in implementation that you'd like to shore up in the near future?

[00:48:58] SR: I think we’re always looking to advance the state-of-the-art both in machine learning and in chabot training, right? So we’re always looking to make this easier to do and we’re always looking to make things sort of just work for you. So if you want to specialize a chatbot, it should be a very short command for you to sort of take one of these existing pre-trained models out of the box and train a new one and find tune it for the specialized behavior. But the neural nets of today may not be the neural nets of tomorrow, right? So you want to remain flexible in sort of what are the different behaviors that we support? And especially what do we support out-of-the-box? And it’s just sort of keeping up with that.

I think there's also always advancements in terms of how do you scale these things. We watched the size of these neural nets increase from a couple million parameters up to billions of parameters over the past couple years. And so there's been a lot of engineering that's had to be done on doing this. But I don't see that curve stopping anytime soon. I think in the next couple years, somebody is going to be training a trillion parameter neural network. And so now how do we keep up with the engineering that's necessary for that sort of huge scale? These are always things that are on my mind; flexibility, future- proofing and scaling.

[00:50:33] JM: Yeah. Talk a little bit more about that. First of all, explain why you would want more and more and more parameters? And second of all, why is that hard to scale up the framework for?

[00:50:49] SR: Yeah. We've been exhibiting a really amazing trend in natural language processing where these high parameter neural networks are learning faster. They're learning better and they're exhibiting behaviors that their smaller counterparts are exhibiting. Sometimes you find that just by going bigger and throwing more compute at it, the end result is so much more than you would've expected necessarily. And furthermore, there's this interesting trend that's happening in the literature now whereas you scale the thing, it actually learns more efficiently. So if you have a fixed compute budget, I'm going to spend X-credits on AWS or on Google Cloud. Then it actually turns out the most efficient thing you can do is train the biggest model that you can with that compute and train it for less time. This is as supposed to the sort of old idea, which was, "All right, let's train a reasonable size model for as long as we can." As we're scaling, we're finding these sort of amazing cost-efficiencies and these amazing behaviors that are emerging.

Where it becomes tricky is you're sort of bound by the computers that you have, right? How do learn these neural networks that no longer fit in-memory? These days, the biggest bound is often GPU memory and how big are the pieces of hardware, these GPUs that you have access to? In a certain point, you start happening to say, "Okay, I've only got 16 gigs for a GPU here. That can only fit 8 billion parameters on it. What do I want to do? And I got to go bigger than that." You have to start saying, "Okay, let me split this up across multiple GPU's." And then you have communication issues, right? So now we have to synchronize state across all these GPUs or split up the data in different ways. And it becomes really complicated to do this efficiently and retain that speed that you need.

[00:52:52] JM: Okay. Well, Stephen, is there anything else you'd like to add about ParlAI?

[00:52:52] SR: Yeah. I mean, come join us. Come check out ParlAI. We've got great tutorials, quick starts. You can be chatting with a model or training your own in just a few minutes. We've got integration with Google Codelabs. So you don't even need access to GPUs yourself. You

can just sort of start out-of-the-box. And we're open to pull requests. So come join us and make the future of dialogue better.

[00:53:20] JM: All right, Steve. Well, thank you so much for coming on the show. It's been a pleasure talking to you.

[00:53:24] SR: Thank you so much having me.

[END OF INTERVIEW]

[00:53:35] JM: If you are a retailer, a big sales day like Cyber Monday can make or break your business. If you sell accounting software, the tax deadline day is like your Super Bowl. And if you're a sports broadcaster, then the Super Bowl is your Super Bowl. Every company has days or seasons that are more critical than the rest. If your systems are ready for the moments that matter most to you, then you're going to be doing much better. And that's the theme of this year's Chaos Conf, the world's largest chaos engineering conference. Chaos Conf is a free online conference taking place on October 6th through 8th, and you can register at chaosconf.io.

Ever since the first Chaos Conf in 2018, the objective has been to create a community around resilience and SRE best practices. Attendees range from having a decade of experience to those who are totally new to chaos engineer. And this year's keynote speakers are Gene Kim, and has been a guest on the show several times; and Adrian Cockroft, who's the VP of cloud architecture strategy at AWS. There will be 20 sessions for all experienced levels focusing on the practice of reliability, completing the DevOps loop and how to build a data-driven culture of reliability. You can register for free at chaosconf.io and the first 1,000 registrants will receive a limited edition swag pack. Claim yours at chaosconf.io and be prepared for the moments that matter.

[END]