Episode 27: Demystifying Large Language Models for the Future of Work – Part 1

You are listening to the HCLTech Digital Workplace podcast, the place where industry experts, analysts and veterans help us identify, understand, and prepare for the upcoming digital workplace technologies and trends. If you haven't subscribed to the channel already, do it now for regular updates. This episode starts in three, two, one.

TJ – "Hello listeners, a very warm welcome to you all! I am TJ and in today's episode, we will be demystifying ChatGPT and large language models. Our guest today is Varun Singh, the President and Founder of Moveworks. Varun leads the product management, design, implementation, customer success, and professional services at Moveworks. And we're glad to have him on the show.
Varun, thank you for joining us. I'm sure that our conversation is going to be really interesting."

Varun: Thank you. It's always great to be back here, TJ. It's a delightful place.

TJ: So, Varun why don't we start with your journey with Moveworks, your role, and how you've seen the company scale over the years?

Varun: Fantastic. So, thank you for inviting me. It's always inspiring to return to India and see how much the country has grown and changed. And technology and organizations such as HCLTech play a significant role in this.

So, Moveworks was founded in 2016. And today we have around 550 to 600 employees. It's located primarily on the West Coast. And we have offices in India, Toronto, and Austin, as well as the UK and Australia.

When we started Moveworks, we began with a very clear vision of the world. We discovered that there are approximately 800 million knowledge workers on the planet. And they all do really important work. But whenever they need help from the business—help let's say regarding a technology issue, a benefits issue, or a workless service issue, these workers always send a ticket or make a phone call. And resolving these issues takes a few days and becomes costly for businesses to resolve.

The reason is that these businesses have massive service desk teams that resolve such issues, and we felt there was a better vision for the world in which people should be able to get help from within an organization immediately. You know, why does it have to take three days to gain access to a software application when I can order a beer, and have it delivered the same day on Amazon? Right.

That was the main issue we pursued with Moveworks. When we examined, talked to customers, and researched, we discovered that people were sending tickets and describing their problems in natural language, using often symptomatic phrases such as long-drawn descriptions of what was going on.

The actual resolution of the issue took about 15 or 20 minutes, and it was frequently as simple as clicking a button or returning the correct article or workflow and letting the user service solve the problem themselves.

But, the latency of going from an employee filing an issue to it getting resolved was about three days, as these issues would just wait in queues. We tried to sort of figure out why these issues are in queues, and it's because someone needs to read the ticket to figure it out. And that's where people came in. That was an aha moment for us. We were thinking, well, if a machine can read the ticket and understand what was going on, it might be able to resolve the problem. And that led to the evolution of Moveworks, where we now support more than 250 organizations worldwide.

And what we're really doing for these organizations is massively improving the productivity of employees by resolving a significant chunk of their day-to-day work issues instantly. And also free up the capacity for service delivery teams, so that they can leverage their intelligence and wherewithal to do more high-value work than just resolving issues for employees. So that's how Moveworks got started. And, you know, with the advent of large language models that we've been using, it's a really exciting time for us right now. We began using R and Ns in 2018. We've been using Bird. This was the first transformer-based model in 2019, and it is now available in GPT 3.5 and 4.

TJ: Varun, that was really nice to hear, about the journey of Moveworks and what it does. So, my question is about the big bang in the world of chatbots and conversational AI, which started this year. For the first time, we had millions of people experience the power of conversational AI with ChatGPT. But the technology behind ChatGPT has been around for a while. Can you talk a little bit about large language models and why the model behind ChatGPT is different?

Varun: Yeah, that's a super interesting question. By the way, ChatGPT, as a bit of trivia, is the fastest-growing application in human history. It has 100 million users in two months, which is faster than Instagram, Facebook, and TikTok. Any application that you can think of it has grown faster than that. And the application itself is very simple, which is why it is so appealing. The application contains only a large language model and an interface for communicating with a large language model.

And that's all you're doing. You're conversing with the underlying ChatGPT, which is a language model. And it was a big, sort of breakthrough moment that opened people's eyes to what large language models were capable of.

However, if you consider the history of large language models, you'll notice that it all began with the concept of deep learning. In addition, these models can be trained on both structured and unstructured textual data.

The technology has existed for a long time in 2017 or so, in 2016, I think Google invented what was called transformer architecture. It was also a method of training these language models. And, in 2015, really bright people founded OpenAI and started training models using this transformer architecture. In 2019, Google trained a model called Bert. Bert, which was based on the transformer architecture in the previous ten years at Google search itself, was the biggest search quality win because it significantly improved the ability to understand natural language queries.

So it started paying off almost instantly. The transformer architecture was a significant innovation. GPT is an abbreviation for generative pre-trained transformer, which is what the model stands for. It is pre-trained on internet skill data, which includes books, the internet, text, Wikipedia, chat logs, and so on. It's also trained in a generative way, which means it can train on both labeled and unlabeled data sets.

When it comes to building ChatGPT, there are two steps. The first is the pre-training of these models. What exactly is a large language model, conceptually? Let's take a moment to unpack that. So, what exactly is a model in the first place? A model's function is to predict. What the real world should look like if it were represented mathematically. So we've all heard of Newton's laws of motion. They predict how a body will move if kicked, and they have some predictive value. So it'll be true to how the world works.

You have weather forecasting models; will it rain tomorrow? There is some chance involved, but it is a mathematical representation of real-world phenomena. Models are what they are. So, what exactly is a language model? A language model predicts what will happen if I use a phrase and then stop. What word should come next?

This is what a language model predicts. So, for example, when we went to see the Golden Gate Bridge, it was extremely foggy. In the city of San Dash, a language model will predict that it's Francisco. And when it makes this prediction, what it's really trying to do is be true to its training data set based on all of the internet data.

In the case of ChatGPT, it is attempting to predict the next word in such a way that it is true to the data that it is being built on. The first step is the discrete training step. GPT 4 is more truthful to its prior data sets than GPT 3.

However, when it comes to interacting with others, these models are at odds. You know, how do you tell a model to be honest? How do you instruct a model to engage a person and how do you instruct a model to converse with it? As if they have no idea what chatting requires.

Deep Mind invented the next step, which is known as reinforcement learning with human feedback. In the late 2010s, the idea was that Reinforcement Learning (RF) with human feedback encoded behavior into these models. And the way that works is actually quite fascinating. I'll just describe the general concept; you might not get the exact steps, but they show a model like GPT, some chat logs of people chatting with each other, and ask GPT to learn how to do that. If they get a person to chat with GPT, that person actually trains the quality of the chat, so they can save high-fidelity to those chat patterns. Or were you good? Were you truthful? Were you offensive? Were you evil? And this goes through maybe a hundred thousand such things.

But as a result of this, you get a score sheet. Like a report card for how GPT is doing. Then they train a new model called a reward model. And what the reward model is doing is it's learning this chat transcript and how it was graded, such that when if you show it any chat transcript, it can give it a score.

Then, they instruct the newly trained chat GPT model to start chatting with itself, such that this model gives you really high points. So the reward model keeps scoring how GPT keeps trying to improve as it wants to achieve a high score on how this reward model is cheating it.

And then, after billions of conversations with itself, it reaches this point where the reward model has given it a high score, and the model is ready. That was the second step, I believe, between the pre-training and the behavior encoding steps, when chatGPT was born as a model. That's a combination of two things: pre-training and reinforcement learning.

TJ: This is amazing stuff. And what are the kinds of things that ChatGPT is good at? And what are some of its weaknesses?

Varun: ChatGPT is a remarkable achievement for humanity. It excels at a few things, which I believe deserve to be mentioned. For starters, it is a single model that excels at a wide range of tasks. It can correct your spelling. Previously when you had very specific models for this stuff. Now it's one model that can do a lot of these things

The second is that it just works with human instruction. You give it human-readable instructions rather than code. And it follows those human-readable instructions. It's excellent for chatting.

And, if you look at some of the use cases, I sometimes refer to it as the ESG of generative AI. It excels at keyword extraction. If you give it a large chunk of text, it will extract keywords. It

excels at summarizing concepts. You give it a long piece of text to summarize. It's very effective at generating, give it a few prompts and ask it to write a poem.

I have a six-year-old daughter, and one of my favorite things is that I can ask it to write jokes for her, and it does an excellent job. However, when it comes to its weaknesses, I believe that the way people perceive it is not necessarily its weakness. They're like, "Well, that's really useful."

It's extremely powerful. How do I make it useful? What prevents it from being useful in its current form? And there are a few things to consider. One, it doesn't have any fidelity to factuality. It's not connected to a database. It's not connected to any source of truth or source of information. It essentially provides you with answers based on some calculations.

It's not too different from the human brain in that regard. So, tell me, how far is Delhi from Tokyo? You'll respond because you know what the world looks like. You have a representation of the world and you'll say, 3000 miles or a seven-hour flight or something like that. But instead of opening a dictionary or measuring on a map to answer that question, you used your memory. That's sort of what ChatGPT does.

TJ: Essentially bringing it down that I know what distance is, I know where these two faces are on a map. I know the scale.

Varun: Yes, but what people don't fully understand about ChatGPT is how it represents the world. Because it is essentially learning the world's representation from language patterns. As a result, it occasionally makes up things. If you're trying to make it useful in the enterprise, you'll need it to integrate with your business systems for it to be useful.

It does not have those integrations built in. It must also be somewhat secure. In some ways, it is learning from everything you share with it. So how do you make sure it's learning the right things and not learning things you don't want it to learn, and that there's no data loss?

So, what matters here is how you deploy it. Let's say you build a big application on something like a ChatGPT. To manage that application, you need analytics, you need reporting. It doesn't have these features because it's pre-trained on internet-scale data. It doesn't have fidelity to your dataset.

As your dataset is not available on the internet, it may not perform as well in the enterprise setting as it does outside. For example, if you ask ChatGPT, where can I get a loaner? ChatGPT would assume you're talking about a car. Because on the internet, when people talk about loaners, they're talking about, Hey, my car's in for repair. I need to get a loaner. But if you ask in the enterprise setting and someone says, Hey, where can I get a loaner? They're discussing a loaner laptop. Because they've either misplaced their laptop or it has stopped working. And these are two distinct data sets.

Enterprise data will understand what a loaner means in the context of an enterprise, whereas GPT doesn't, so it's not so much the weaknesses. I would say it's more a way to think about how you deploy GPT meaningfully in an enterprise. You need to still augment it with a lot of things to make it useful.

TJ: So, like a fresher or a new employee, it has to get well versed with the terminologies that are used on a daily basis.

Varun: That's one example. It also needs to understand the concept of fine-tuning, which is used in the machine-learning world. This model comes pre-trained with certain weights.

For example, we have a petabyte of enterprise data and regularly fine-tune models like GPT to improve their performance. We've been using large language models for a long time, and this fine-tuning step increases the fidelity of this data set to the enterprise world or the fidelity of these models to the enterprise world, and it's sort of like a fresher or new grad joining, they're now sort of getting versed with this business lingo, and can work with it more successfully.

Another example is grounding, which is if you are a fresher, we can stick with that analogy.

If you think about large language models, the way they work is you prompt them. You give them a prompt and they do something. Therefore, the quality of how good these models are at performing a certain task depends on the quality of the prompt.

So, if you give it a good prompt, it'll do a good job. A lot of effort that our applications and our engineering team do is when our application calls these large language models. It's trying to engineer a prompt that results in a better performance from this model. What does that mean?

If you think of a fresher and ask them to do a new job, but give them three examples, it's as if you're saying, "I had something like this and this is what we did." And go ahead and do the same thing for this new assignment. A newcomer has a much better chance of success, if you did not provide them with those examples. And what our applications do is look at enterprise data to generate those examples that are embedded in the prompt when calling GPT 3 to reproduce a specific task where we've given it examples or more context about what needs to happen.

And now all of that happens at the engineering stage. So that's another way in which you can make these models useful in the enterprise.

This is the end of the first part of our interaction with Varun. We'll continue this discussion in the second part. See you soon.

This episode of the HCLTech Digital Workplace Podcast is over. But be sure to subscribe for more insights on how to identify, understand, and prepare for a world of possibilities around

new and upcoming digital workplace technologies and trends. Don't forget to rate and review this episode, so we keep bringing you the most relevant content.

Thank you for listening.