EPISODE 583

[INTRODUCTION]

[0:00:00.3] JM: Algorithms for building neural networks have existed for decades, but for a long time neural networks were not widely used. Recent changes to the cost of compute and the size of data available have made these neural networks useful. Our smartphones generate terabytes of useful data. Lower storage costs make it economical to keep that data. Cloud computing democratized the ability to do large scale machine learning across GPUs.

Over the last few years, these trends have been driving widespread use of deep learning, in which neural nets with large series of layers are used to create powerful results in various fields of classification and prediction. Neural networks are a tool for making sense of unstructured data, text, images, sound waves and videos.

Unstructured data is data with high volume, or high dimensionality. For example, an image has a huge collection of pixels and each pixel has a color value. One way to think about image classification is that you are finding correlations between all those pixels. A certain cluster of pixels might represent an edge. After doing edge detection on pixels, you have a collection of edges and then you could find correlations between those edges and decide where those edges abstract and you build up higher and higher levels of abstraction.

Yinyin Liu is a Principal Engineer and the Head of Data Science at the Intel AI Products Group. She studies techniques for building neural networks. Each different configuration of a neural network for a given problem is called a topology. Engineers are always looking at new topologies for solving a deep learning application, such as natural language processing. In this episode, Yinyin describes what a deep learning topology is, and she describes topologies for natural language processing.

We also talked about the opportunities and the bottlenecks in deep learning, including why the tools are so immature and what it will take to make the tooling better. Full disclosure, Yinyin works at Intel and Intel is a sponsor of Software Engineering Daily.

[SPONSOR MESSAGE]

[0:02:33.9] JM: At Software Engineering Daily, we have user data coming in from so many sources; mobile apps, podcast players, our website, and it's all to provide you our listener with the best possible experience. To do that, we need to answer key questions, like what content our listeners enjoy, what causes listeners to log out, or unsubscribe, or to share a podcast episode with their friends if they liked it. To answer these questions, we want to be able to use a variety of analytics tools, such as Mixpanel, Google Analytics and Optimizely.

If you have ever built a software product that has gone for any length of time, eventually you have to start answering questions around analytics and you start to realize there are a lot of analytics tools.

Segment allows us to gather customer data from anywhere and send that data to any analytics tool. It's the ultimate in analytics middleware. Segment is the customer data infrastructure that has saved us from writing a duplicate code across all of the different platforms that we want to analyze.

Software Engineering Daily listeners can try segments free for 90 days by entering SE Daily into the how did you hear about us box at sign-up. If you don't have much customer data to analyze, Segment also has a free developer edition. But if you're looking to fully track and utilize all the customer data across your properties to make important customer-first decisions, definitely take advantage of this 90-day free trial exclusively for Software Engineering Daily listeners.

If you're using cloud apps such as MailChimp, Marketo, Intercom, Nexus, Zendesk, you can integrate with all of these different tools and centralize your customer data in one place with Segment. To get that free 90-day trial, sign up for segment at segment.com and enter SE Daily in the how did you hear about us box during signup.

Thanks again to Segment for sponsoring Software Engineering Daily and for producing a product that we needed.

[INTERVIEW]

[0:05:03.8] JM: Yinyin Liu is a Principal Engineer and Head of Data Science at the Intel AI Products Group. Yinyin, welcome to Software Engineering Daily.
[0:05:10.9] YL: Thank you for having me.

[0:05:12.3] JM: You've worked in machine learning for over a decade, which is a long time given that there's a lot of new people entering machine learning today. How have you seen the field change over that period of time you've been involved?

[0:05:26.4] YL: Yeah, it's definitely a very interesting experience in the last decade. At beginning, that there are a lot of new ideas and gradually these new ideas are trying to extract information from the data, to allow a computer program to learn. A term, this technology called machine learning and machine learning actually touch upon a lot of different disciplines and area around 2012, this new discipline called the deep learning started leveraging a lot of the new resources data set and the computing infrastructure.

As everyone can see that a deep learning has become a very prominent discipline within deep learning. These days, researchers are having a lot of momentums and a lot of collaborations throughout the open source community and open research community working on a lot of the deep learning problems, altogether to push the field forward.

[0:06:27.6] JM: The deep learning techniques that people are using today were around a long time ago, but it seems there were a number of prerequisites that actually needed to be built in order to make use of these techniques. I think those prerequisites were you needed cheaper data storage and you also needed people generating that amount of data. Now you've got smartphones and sensors and people doing internet searches. You've just got more engagement with the digital world that has produced all this data and you've got places to store it, thanks to cloud computing.

You've got all these lower level processing primitives for doing high-volume data science, like Hadoop and Kafka and Spark. From my perspective, it was those prerequisites that really allowed machine learning to take off. Is that consistent with your experience?

[0:07:25.5] YL: Yeah, you're exactly right. Machine learning, or deep learning sounds like a simple term, but I actually touch upon a lot of technologies and products, or opportunities going into the enterprise use cases, the consumer use cases, altogether make the current state of the field possible. A lot of the technology you just mentioned that really impact how a technology company really to provide the products, or a portfolio of products to support all kinds of deep learning and machine learning use cases.

Coming from the data collections and gathering more resources for training data and testing data, that touch about a lot of products and that how you really allow users to provide their input, how you gather resources from many different sensory inputs, then the data storage, data transportation. Then eventually you come to the compute side, even with the similar machine learning algorithms based on how that data pipeline is structured, then it plays a very different requirements of how that product should handle that data and how to handle the compute. Really that in order to enable all these ideas and the technologies into something that can impact a daily usage, it requires a whole set of products and technologies.

[0:08:59.2] JM: We've certainly covered the software components of those in detail. You've spent some time in electrical engineering and you work at Intel. Intel’s core competency started in the electrical engineering side of things and has worked up the stack towards having competency in software as well. How important is that knowledge of hardware when it comes to developing machine learning tools?

[0:09:27.8] YL: Yeah, so earlier you also mentioned that nowadays, that machine learning and deep learning are possible because of the newer hardware support on this type of compute or workload. Definitely, in order to make machine learning solution working or create some good analytic solutions that you need a very good compute.

There are a variety of different hardware options available these days in the market for people to choose what type of hardware is more suitable for each one’s individual use case. Then understanding what hardware you have and what are the constraints you have, definitely create a lot of insights for data scientist and/or machine learning engineers to structure the set up in order to create such a solution.

Working at Intel for myself were part of AI product school. The AI products as I mentioned, it touched upon a whole portfolio of a different type of hardwares and softwares and many different other products. Working in a group that with all these different hardware activities and development going on, that allow me, or the background in the hardware and electrical engineering allow me to talk to a hardware engineer more easily
Even though I'm standing from the algorithm perspective, or from the system perspective, I can post a question or create the requirements, or project on the future hardware trends and working with hardware engineers very closely.

Similarly, the background in software engineering provides a similar framework as well, so that using – being very familiar with all kinds of deep learning frameworks and the software allow me to quickly structure a certain data science models and create a certain solutions. Also once I see different requirements that the current deep learning frameworks are not supporting, I will be working with the teams that are developing frameworks, software together to make the framework better to support newer and more interesting models.

[0:11:54.2] JM: What are the operations that hardware needs to optimize for? Because I know that a lot of machine learning models boiled down to processing large sets of matrix operations. What needs to be optimized for at the hardware level?

[0:12:11.0] YL: When we look at all these different new deep learning models to solve a computer vision applications and NLP applications, working for some learning so and so forth, each of these different areas requires a set of similar relevant, but also different type of neural network architectures.

These neural network architectures will boil down to lower-level computing motifs. For example, in computer vision, convolution is a very commonly used computing components. Bash normalization is also another very commonly used to computing components. Similarly in NLP area, that we have the current neural networks, or the newer ideas are temporal convolution neural networks.

Then in robotics, where reinforcing the learning, there will be other computer motifs such as sampling algorithm. All these different type of architectures boil down to a set of overlapping, but and some are very different computer motifs. For hardware to best support all these different type of applications or workload, we need to make sure that the hardware support these features not only from a functional perspective, but also from a performance perspective, so that when people use these hardwares that they can get good accuracy, but also can do training quick enough and it can also run a trend model in inference mode fast enough before their applications.

[0:13:55.6] JM: I see. You have been building tools for these different deep learning domains, like robotics, NLP, computer vision. You have to interface with all these different teams, different people at the software layer, at different – you probably have to interfaces with mathematicians, certainly some hardware engineers. From your experience interfacing with all of these different people, what are the classes of problems that you think neural networks and deep learning are useful for? Is there is there a broad swath of problems that we're going to see advancements in?

[0:14:40.2] YL: Yeah, sure. First thing comes to our mind is that nowadays with typical deep learning algorithms is mostly about converting a problem into numerical representation and try to compute, try to extract the features and a compute output using these numerical representations over vector matrix representations of data. That any type of data, as long as there is a good way to represent them numerically and to do these parallel type of compute, then it will help us to solve the problem.

For example, in computer vision the very natural way of a representing image is through the pixel values. In natural language processing, there are a lot of research work to be done in order to find out the optimal way to represent attacks using numbers as well. There are works that are trying to represent texts on work levels, or on sentence levels, or even on character levels, convert them into vector representation.

Similarly, that in robotics and reinforcement learning, the inputs, the sensory inputs into the robot agent that whether it is images, or audio, or some state representation. Because essentially, all these different type of inputs can be represented as dense vectors, then it really allow a neural network to efficiently to process them and then leveraging the hierarchical structure of the neural networks, it will learn how to best represent the features and information of different levels of abstraction, in order to achieve the task at hand.

[0:16:37.6] JM: Those different domains, they sound like most of them have some lack of structure and much of the topology of a deep neural network is about bringing structure to the unstructured data. You have, in the case of computer vision, the earlier layers in a neural network are about convolving to find edges within an image. Then you can use those edges that let – a set of edges is a more structured data set than a collection of pixel values.

I mean, a collection of pixel values does have a sense of structure, but it's such a fine-grained structure and there's so much density to it that it's hard to know what to do about it, so you have to reduce the dimensionality, right? Or maybe not reduce the dimensionality, reduced the – I guess in that sense, you'd be reducing the cardinality –

[0:17:37.9] YL: That's right.

[0:17:38.9] JM: - of the number of data points. Then similarly with NLP, you've just got this unstructured text and you need to add context to it, and there's things like – well, I guess I'm less familiar with NLP. We did a show about Word2vec, but I'm not sure what you can do with a large blob of unstructured text.

Yeah, maybe – I mean, one thing I'm curious about, like if you take one domain, one domain of unstructured data, like the image example, and there are advancements in the image recognition field where you have – where we can make sense of us a large set of pixels that have RGB values, how well do advances in image recognition translate to another domain involving unstructured data, for example NLP?

[0:18:29.3] YL: You're exactly right. It's a good question that as you described it that in image case that the network, the goal of the network is try to convert more primitive signals with a lack of structure, to a more structured, or abstracted representations. These type of features, or functionalities transfer pretty well from domain to domain, that in natural language processing case that the first level of input is a usually character level representation, or word level, or sentence level.

Then you can gradually build up that when you look at multiple characters, or multiple words, what are some of the high-level features that you can extract? Then gradually build up from group of words into group of sentences, or group of paragraphs. Also in language, there is a unique feature about temporal continuation, in the sense that one sentence after another usually have a lot of the semantic meaning built-in, in that sense that when we try to train a network that can handle such temporal transitions from words to words, or from sentence to sentences, the network itself is able to correlate more higher-level temporal features and to really extract the information from language data.

[SPONSOR MESSAGE]

[0:20:14.6] JM: Azure Container Service simplifies the deployment, management and operations of Kubernetes. Eliminate the complicated planning and deployment of fully orchestrated containerized applications with Kubernetes.

You can quickly provision clusters to be up and running in no time, while simplifying your monitoring and cluster management through auto upgrades and a built-in operations console. Avoid being locked-in to any one vendor or resource. You can continue to work with the tools that you already know, so just helm and move applications to any Kubernetes deployment.

Integrate with your choice of container registry, including Azure container registry. Also, quickly and efficiently scale to maximize your resource utilization without having to take your applications offline. Isolate your application from infrastructure failures and transparently scale the underlying infrastructure to meet growing demands, all while increasing the security, reliability and availability of critical business workloads with Azure.

To learn more about Azure Container Service and other Azure services, as well as receive a free e-book by Brendan Burns, go to aka.ms/sedaily. Brendan Burns is the creator of Kubernetes and his e-book is about some of the distributed systems design lessons that he has learned building Kubernetes.

That e-book is available at aka.ms/sedaily.

[INTERVIEW CONTINUED]

[0:21:49.8] JM: NLP and image recognition, these are two fields that have been studied deeply. They've been studied so deeply that there are experts, such as yourself who has to have studied both things. You can draw on your expertise in NLP when you are doing something in image recognition. Deep learning is this hammer for which there are many, many nails. Any domain with lots of unstructured data, you could take something like the air quality and try to train what kinds of air quality do athletes perform the best in, right? That's something that I'm certain there would be value in studying that.

If you're just tackling a new problem like that, is it is it trivial to figure out the fields, the topological methods that you can borrow from other domains, or does it take a matter of iteration and research? If you're walking into a new field with a lot of unstructured data that you're trying to build models for, what's the methodology for pulling topological strategies from other domains?

[0:23:10.3] YL: Good question. Definitely that in order to build a model that works for a practical use case, it has challenges in many different aspects. Definitely that being familiar with multiple domain topologies covering computer vision, NLP, on so on and so forth, help us to be more familiar with other options we have. Throughout the experience on working with the computer vision models, or NLP models, we get to build up this experience on what type of topologies would work well with what type of data? Would it work for supervised learning, or unsupervised learning, or unsupervised pre-training? There are a lot of ideas that we build up throughout the experience on these different models that we can try and borrow from the past experience.

One thing that is very interesting to see these days, specifically related to deep learning advances is that some of the newer deep learning advances that starting from computer vision models, if we tried it out in some of the existing NLP tasks, they actually also create better performance, or improve the efficiency as well.

For example, that the highway network, or dense connections, these ideas were originally mostly coming from solving computer vision bottlenecks. Nowadays that we can see that they improved the NLP model performance as well. When we look at from a new use case perspective, first of all, we need to understand on a high-level what type of data we're dealing with. Is it mostly a special type of data, like images, or it has a lot of a temporal component in it, like a language or time series?

Then based on the high-level understanding on the nature of the data, we can look at all the options in terms of topologies and the techniques that people have shown on some of these typical applications, and then try out the models that you wanted to build for this particular problem.

Another layer of challenge is that the data is often being very messy, and not only being unstructured, but also very messy. In order to construct the features that can go into the neural networks, it takes a lot of understanding about the domain and about the use case itself to see that what are the different sources of data that we can pull together, in order to at least structure some features, in order to feed into the neural networks?

In this type of experience, often that Intel that the data scientist and the machine learning engineers are working with our partners and customers very closely, so we get to we get access to a lot of domain knowledge as well, in order to really understand how we can structure the data. Then based on our own experience, to build the topology that works for the problem and eventually try to build up a solution by using the right software choice and the right hardware choice.

[0:26:45.2] JM: The topology is the different layers of the neural network and how data passes through those different layers and feeds data, feeds updates back to the previous layers, is that right?

[0:27:02.5] YL: That's right.

[0:27:03.6] JM: Okay. Why are there so many different – this might sound naïve. Why do different neural networks have different topologies?

[0:27:14.7] YL: Yeah, it's actually a pretty important question, because as you said earlier that a layer, the role of a particular layer in a deep neural networks is to process the data from one level to the next level, but also take some error input based on the optimization on the top level, take that error input and to adjust the weights within its own layer.

There's a seemingly simple process. Actually on a numerical perspective, it has a lot of challenges, because sometimes if your network is very deep, then it's possible that the error signal you receive being back propagated is through many other layers is losing the information that you can actually learn from you as one of the layer inside this deep neural networks.

A lot of the techniques and the deep learning advances are really developed to tackle challenges around this area. For example resonant was developed in order to better provide ways to understand whether a layer is needed, or help the gradient, or error information fact propagate through the entire network. Similarly highway network was constructed with a similar goal. Dense network was also created, so that as a layer itself not only it can receive input from adjacent previous layer, but also it can receive input from several other previous layers, so that to make sure that if the adjacent layer doesn't do a good job to provide some input, there are other layers that provide additional input to help for each of the layer to learn.

These are just the layer topology perspective besides the topologies, then there are other techniques in terms of designing the optimization techniques, to also improve the learning capabilities of each individual layers. Then topology itself, not only dealing with the learning perspective, but also need to deal with what is the best structure to deal with the data itself.

When we deal with special data versus temporal data, that it replace different requirements what the layers should be doing. For example in recurrent neural networks, the gradient or error message challenge we talked about earlier in image case actually become more prominent, because every layer itself is going through this time unrolling process.

Imagine this layer not only need to receive input and send out output, but within itself when these two unroll iterations for example 50 or a 100 steps to really ingested this time series, or this language data. In the process of doing this unrolling, it definitely plays additional challenge, because your original input get lost through the unrolling process. Similarly in the backpropagation process, that the error message that you need to learn from may also get lost throughout the process.

That's why that there are many different topologies and techniques are designed in the past few years and even very actively these days try to make the network to learn better and in a more robust way.

[0:31:05.7] JM: I want to dive into a specific use case for people to understand how you think about deep learning. The use case that I want to talk about is object detection. Deep learning topics are always hard to discuss on the podcast, and I think I'm gradually getting better at overseeing discussions of them, but we'll go as deep as we can.

Let's start with an explanation of what object detection is, because this is a sub-topic of image recognition tasks. There's object classification and there's also object detection, what are the differences between these two problem sets?

[0:31:51.3] YL: For object classification, the goal is to predict the category, the overall class or category of the image. Then for example, a network will take image as input and the output will be just a label saying it's cat, or dog, or it's a house. For object detection, or sometimes called the object localization, it will take the input, the image as input to the network. The output of the network has two different type of output; one is a group of bounding boxes the network finds from this input image. Associated with each of the bounding box, the network will provide a label.

As you can see that combining the bounding boxes with the label, that the network is able to tell very precisely that in this current image, there will be a cat at certain location and that there will be a car at a different location.

[0:33:00.4] JM: You're trying to detect where in an image an object is, as opposed to just the binary is this a cat? You're trying to find exactly where in the image the cat is?

[0:33:16.7] YL: That's exactly right.

[0:33:18.1] JM: I think an object detection model is good to run through, because it explains the importance of pre-trained layers. Image recognition neural networks can have these pre-trained layers. Explain what a pre-trained layer is.

[0:33:34.4] YL: For a topology, that originally used for image classification. Many of the layers have been trained on a big data set. For example, image net that consists of a thousand causes and a million images. Throughout such a training process, these layers have been optimized to represent the features on different levels. After such an image classification training, we wanted to reuse what we have learned, what the network has learned. By taking out these layers and put them into a new network architecture that build based upon these set of layers to do further things. Further things including definitely object detection.

[0:34:22.4] JM: Definitely. The idea of a pre-trained layer is almost like a module that you're importing from somebody's previous work on some deep learning project. In the case of object detection, you can utilize that pre-trained layer to give you a head start, you don't have to rebuild your entire model from scratch.

The other important parts of an object detection process are to create a feature map and then to define regions of interest within your image. Then you have to define where the regions of interest map to features. Can you explain briefly what a feature map is and what a set of region of interests are, and how those two things get unified in order to define where in an image different objects lie?

[0:35:24.2] YL: Sure for a feature map, that as you can consider that a network with several layers, with a network with the several layers, which are good at extracting image features at different levels of abstraction. At the beginning that it would be more detracting edges, or small features and gradually you started building up a higher level feature. A certain point in the network that a layer of neurons take the input and create a certain reaction were activations to these set of input.

Then that output activations is usually considered as a feature map. Such a feature map that represent many different interesting information, so for example, if the neurons inside such a deep networks is learned to detect a certain high-level features, such as detecting eye, or a part of a car, then it really shows that now with this current input image, whether such a feature exists inside your input image.

Whereas, the feature map it also shows that where in the image has interesting features, such as if you have a lot of high frequency features with a lot of textures, or with the change of a background, or with a certain contour of a certain object. All these things will trigger the layers to react in a very different way.

To train a network to detect a region of interest using the similar idea that a region of interest is often defined as area with a certain object in it. Or you can consider with a certain background, and a part of that image, a part of that object is more distinct from the background itself. With such distinctive features, it can trigger the network to react in a way that this area is a region of interest by telling the network which part of the feature map is more active.

Looking at the more active part of the feature map, then it gives us this region of interest. Then in object detection network, there could be – one type of the object detection network is using an operation called ROI pulling. It pulls features from region of interest ROI into a small map. Now do the image of classification only within the ROI itself, to do the object classification. As you can see that it does this object detection in two steps to detect that ROI first, and then to detect the class within the ROI.

[0:38:27.6] JM: Definitely. I'm glad we ran through the object detection example, because I think it's a tangible example. People can understand images, they can imagine images and certainly anybody that's working on object detection will find it useful to have that explained by you.

I want to talk about NLP a bit now. You've been spending a lot of time in NLP. How does the field of NLP and the topologies of NLP compare to the object detection topology that you just described?

[0:38:59.8] YL: Yes. For NLP models that one distinct factor is about handling temporal data, so that it will definitely borrow ideas from image of classification topologies, but also incorporate the factors that the network itself needs to know how to handle the temporal features itself. I think some ideas that really borrowed from the image domain is things like a skip connections and dense connections. Then it help the network to really create better ways of back propagating the error signals for the network to learn.

Other than that, I think in general convolution networks is a common topology used for image domain as – for image domain. For NLP, that there are a lot of empirical results to show that convolution layer itself can be directly applicable to NLP tasks and create a very compelling accuracy results as well.

[0:40:07.2] JM: What are some of the other trends that you see in the NLP space?

[0:40:11.1] YL: One of the trend that I found is very interesting, is that nowadays, that even though there are differences between the data that it needs to handle, a lot of the technologies transfer pretty well from other domains to NLP. In terms of the topology it changes, in terms of techniques optimizations, in terms of data organization. Also there are a lot of interesting ideas using generative models to handle NLP as well.

I see the way that these days that inside the deep learning, a lot of the advances is triggered by computer vision, or robotics, or time series analysis, data transfer NLP as well. I help NLP to take on approach that is complementary, but also different from the traditional approach as in a way that a traditional approach, that is usually driven by human design rules, or understanding the language itself. These days that was these type of deep neural networks, it helped the NLP space to be more adaptive to the big data trend, and being able to in the performance based on a bigger dataset based on newer technologies. It doesn't really require a lot of in-depth understanding about the language itself before you can create some interesting NLP models yourself.

[SPONSOR MESSAGE]

[0:42:02.5] JM: LiveRamp is one of the fastest-growing companies in data connectivity in the Bay area. They're looking for senior level talent to join their team. LiveRamp helps the world's largest brands activate their data to improve customer interactions on any channel or device.

The infrastructure is at a tremendous scale. A 500 billion node identity graph generated from over a 1,000 data sources, running a 85 petabyte Hadoop cluster and application servers that process over 20 billion HTTP requests per day. The LiveRamp team thrives on mind-bending technical challenges. LiveRamp members value entrepreneurship, humility and constant personal growth.

If this sounds like a fit for you, check out softwareengineeringdaily.com/liveramp. That's softwareengineeringdaily.com/liveramp. Thanks to LiveRamp for being a sponsor of Software Engineering Daily.

[INTERVIEW CONTINUED]

[0:43:09.9] JM: There's a lots of enterprises that have tons of documents and various text formats that they would want to have NLP processing done on. What are some of the low-hanging fruits, the use cases that enterprises can apply NLP to and get significant value?

[0:43:32.8] YL: A couple of interesting applications we see often from our partners and customers are things related to sentiment analysis. An intuitive example for sentiment analysis is that you can look at reviews related to for example a movie. That really help you to understand how the users, reviewers, the audience look at such movie, and by providing a review on different aspect.

Such a sentiment analysis is not only useful for movie reviews, but also is directly related to how you can understand your product adoption, how people react to different aspect of your product. Also for example, in terms of customer relations what are the aspect that customers reacted differently in terms of vocals analysis, then you can see that what are the aspect that usually triggers positive feedback, or negative feedback.

One particular application such as sentiment analysis can actually bring many different business insights and related to the business models or the product that a particular enterprise user is building, then they can use these applications in several different ways. Another prominent use case is document similarity search.

Often that enterprise user has a large collection of documents to manage, to extract the insights and knowledge from. Because of the amount of text and amount of information someone needs to handle, it becomes unmanageable for a human user. With a good NLP application that you can do document and similarly search, you can do relation instruction, and you can do keyword detection. With a set of very related NLP functionalities, you can create multiple applications and to suit your business requirements.

[0:45:55.2] JM: What are you working on – are you working on anything specifically related to that? Is your team working on anything specifically related to those use cases?

[0:46:02.4] YL: Yeah, so at Intel that we have a team of NLP developers and researchers for a long time that we are very familiar with all these a different state-of-the-art algorithms and new topologies and new algorithms. In the process of doing these research and development, we feel like we could have put a lot of these NLP components, or applications, or modules into a more comprehensive stack.

We can look at this as a stack of NLP capabilities. For internal development that people can share what are the individual modules that each one is working on, but also many of these modules that can be reused and combined in many different ways to create applications, or solutions.

We create these set of NLP modules as an opiate library, and we're hoping to make this available to the open source community sometime soon, to create more collaborations and a research with the overall community as well.

[0:47:20.4] JM: To zoom out, I want to talk about more developer-oriented subjects. For example, the tooling around deep learning; today it feels like it's insufficient. It's like, if I want to stand up a web application, I could use Ruby on Rails and it's not very hard. If I want to build a deep learning model for something, it's still quite difficult, there's a lot of domain-specific knowledge I need to know. I need to know about deep learning topologies. When is the tooling going to get better? Or is it going to get better is it always going to be this hard?

[0:48:04.4] YL: Well, deep learning is fast-evolving field. It is exciting on one hand for researchers, but also very challenging for developers who are building tools, because as soon as you provide a very good support set of functionalities, or components, then there are new ideas and new topologies that your tool may not support.

I think in general, that these days, that we do see that there are tools that are very tailored for beginners that you can start by calling some high-level APIs and quickly create a neural networks, and that there are aspect of the tools to help you to ingest a data set, so that you can build a model for a particular problem.

Overall, there are a lot of efforts going on to help the user to learn and practice and understand the deep learning and become a deep learning data scientist. A good thing is that all these tools are all open source, and we have a very active open source community that the developers are supporting users and the users can even make contributions to the tool itself, if you think that there are ways to improve that kind of user experience.

Definitely, it is because the fact that the community is very active, there are a lot of new research ideas, then there will be always the area that the tool can improve. From my perspective at Intel, that we did try a lot of ways to make sure that the initial user experience is as smooth as it can get.

Also inside the tools, we try to provide a lot of tutorials and examples. Usually when you start using a new framework that by going through some of the beginner's examples, usually people get a good sense on how to get started using this.

[0:50:18.3] JM: I think even just the hierarchy of abstractions, even if we don't have the tools to work with those abstractions at the right – from the from the developers' point of view correctly – you've drawn out this hierarchy. At the lowest level, you've got the hardware. Then on top of the hardware, you have the frameworks and then you have the neural network layers that you can build within the frameworks. Then those layers get composed into a topology, and we've talked about some different topologies, like things that you would build for object detection, or NLP use cases.

Then the topologies can be abstracted into these higher-level components. You could imagine a higher-level component for object detection. Then those higher-level components can be abstracted into applications. Then the applications, we know how to build those, we know how to build applications from components, from APIs and we know how to direct those use cases.

That seems like a bright future. Once we have the application level building blocks, things will get much easier.

[0:51:30.4] YL: For sure. Yeah, so I think these days that we do see that in terms of toolings, there are several layers that we can go beyond besides deep learning frameworks. As we were talking about there are layers and topologies components applications, these are the layers and stacks sitting on top of some of the more fundamental deep learning frameworks. All these layers will help users to get started based on which level of abstraction, or level of flexibility that they want for their own particular use case. Definitely that the more component we can put into these different levels of the stack, really can help and serve the need for many different developers for their own applications.

[0:52:24.9] JM: You are part of the partnership on AI, which is – Well, why don't you explain to me what the partnership on AI does?

[0:52:33.3] YL: Partnership on a AI is a nonprofit organization found in 2016. The organization is established to study and formulate some best practices AI technologies and help the public understanding of AI advances. As you know that with the fast-evolving field of AI, there are a lot of concerns and questions on the overall direction of AI technology, what type of benefit and impact it will really bring to the society.

The organization tries to bring people from different disciplines and create a platform, or forum for these type of very open discussions and engagement, to understand what could be the AI influences on people and society, and how, as an organization how partnership on AI can have other partners work together to make sure that the AI technology is evolving in a direction that is beneficial for industry and for public and for society and government as well.

[0:53:51.4] JM: These are important issues. I just finished reading a book by Max Tegmark, where he was talking about some of these AI safety issues. It's pretty hard to reason about, because it's so speculative at this point, but it doesn't mean that we shouldn't be thinking about it.

[0:54:09.3] YL: Yeah, exactly. I think often, the case as AI developers and practitioners, we might be very focused on solving the problem itself. People from very different perspective that can actually see the impact to beyond the problem that we intend to solve. Understanding what could be the potential impact going into the society and people and try to make sure that the technology is building in a way that is really beneficial for short-term and long-term impact.

Even for the topic of how to make sure AI is beneficial to all these things, it's a very complex topic. It has several different aspects that people can try to start to address. Things related to safety critical AI applications, that touch upon things like autonomous driving is a very good example how we can make sure that the AI system building into the autonomous driving solutions is really addressing a lot of these safety issues.

There are topics related to fair transparent and accountable AI, so that when we build AI system, we need to make sure that it doesn't create unnecessary bias, or unintentional bias. The AI system itself can be open and transparent, for people to understand how to trust this AI system and how to use this in a very fair way.

There are a lot of other topics that being structured within partnership on AI as well, things like AI labor and economy, or AI for social good. These are all very important topic. Of course, as I said overall, it's a complex topic so that we have 54 partners at this point within partnership on AI, so all these people coming from very different disciplines are trying to look at all these different aspect and think of ways, or projects that we can work very closely on.

[0:56:35.2] JM: I think you're going to be at the Intel AI DevCon, which I'll be attending as well, in I think a few weeks. Wat are the themes that you're expecting to see at that conference?

[0:56:47.8] YL: Yeah, Intel AI DevCon is May 23rd. The trend that I'm expecting and very excited to see that there will be people from several different disciplines within AI, from data science and machine learning engineering, from application development and from research get together to share their latest work and share their latest perspective. Also the goal to have a developer conference is that we wanted to show that all these work that we did in the recent time, that how really bring more ideas and options for the developers, for them to use these technologies and solutions we build to solve their real-world problems as well.

We're going to talk about a lot of these products that Intel is building in terms of AI framework softwares, in terms of different hardwares. Also, to showcase some of the use cases we worked on with our industry and academic partners. For example, there will be our customers and partners that present at DevCon as well, to talk about some of the healthcare use cases, genomics use cases, or scientific simulation type of use cases that we work together on.

[0:58:21.8] JM: Yinyin, thank you for coming on Software Engineering Daily. It's been really great talking to you.

[0:58:25.0] YL: Yeah, thank you for having me. Bye.

[END OF INTERVIEW]

[0:58:30.5] JM: GoCD is a continuous delivery tool created by ThoughtWorks. It’s open source and free to use and GoCD has all the features you need for continuous delivery. Model your deployment pipelines without installing any plugins. Use the value stream map to visualize your end-to-end workflow. If you use Kubernetes, GoCD is a natural fit to add continuous delivery to your project.

With GoCD running on Kubernetes, you define your build workflow and let GoCD provision and scale your infrastructure on the fly. GoCD agents use Kubernetes to scale as needed. Check out gocd.org/sedaily and learn about how you can get started. GoCD was built with the learnings of the ThoughtWorks engineering team, who have talked about building the product in previous episodes of Software Engineering Daily, and it’s great to see the continued progress on GoCD with the new Kubernetes integrations.

You can check it out for yourself at gocd.org/sedaily. Thank you so much to ThoughtWorks for being a long-time sponsor of Software Engineering Daily. We’re proud to have ThoughtWorks and GoCD as sponsors of the show.

[END]