EPISODE 712

[INTRODUCTION]

[0:00:00.3] JM: Mapillary is a company that processes high volumes of images to develop a labeled 3D model of the physical world. Mapillary's APIs allow developers to build applications that are aware of stop signs, buildings, streets, trees and other physical objects in real-world space.

The potential use cases for Mapillary are numerous, ranging from self-driving cars to augmented reality. We can now build a 3D model of the real world and expose APIs to that 3D model of the real world. It's not a perfect representation of reality, but it is much better than we had just a few years ago. What has changed? How have the tools advanced, such that we are able to build an API for accessing information about the physical world around us?

Mapillary is possible because of a combination of modern developments. High-quality smart phone cameras allow users to crowdsource images of the world around them, cloud computing allows for cheap workload processing, newer computer vision techniques allow 2D images to be stitched together in a 3D representation. Deep learning architectures improve the classification and segmentation of objects in an image.

Peter Kontschieder is the Head of Research at Mapillary and he joins the show to talk about the technologies and research that has enabled Mapillary to build a futuristic business; an API for accessing information about the physical world.

Software Engineering Daily is looking for sponsors. If you're interested in reaching over 50,000 developers, you can go to softwareengineeringdaily.com/sponsor to find out more. You can send us a message. We'd love to hear from you. If you're an engineer working at a company that is marketing to developers, or if you are hiring developers, if you tell your marketing department or your recruiting department about softwareengineeringdaily.com/sponsor, that is one way to help us out.

Thanks for listening and let's get on with the show.

[SPONSOR MESSAGE]

[0:02:16.7] JM: Managed cloud services save developers time and effort. Why would you build your own logging platform, or CMS, or authentication service yourself when a managed tool or API can solve the problem for you? How do you find the right services to integrate? How do you learn to stitch them together? How do you manage credentials within your teams, or your products?

Manifold makes your life easier by providing a single workflow to organize your services, connect your integrations and share them with your team. You can discover the best services for your projects in the Manifold Marketplace, or bring your own and manage them all in one dashboard. With services covering authentication, messaging, monitoring, CMS and more, Manifold will keep you on the cutting-edge, so you can focus on building your project, rather than focusing on problems that have already been solved.

I'm a fan of Manifold, because it pushes the developer to a higher level of abstraction, which I think it can be really productive for allowing you to build and leverage your creativity faster. Once you have the services that you need, you can deliver your configuration to any environment, you can deploy in any cloud. Manifold is completely free to use. If you head over to manifold.co/sedaily, you will get a coupon code for $10, which you can use to try out any service on the Manifold Marketplace.

Thanks to Manifold for being a sponsor of Software Engineering Daily. Check out manifold.co/sedaily. Get your $10 credit, shop around, look for cool services that you can use in your next product or project. There is a lot of stuff there and $10 can take you a long way to trying a lot of different services. Go to manifold.co/sedaily and shop around for tools to be creative. Thanks again to Manifold.

[INTERVIEW]

[0:04:30.9] JM: Peter Kontschieder, you are a Head of Research at Mapillary. Welcome to Software Engineering Daily.

[0:04:35.3] PK: Hi Jeff. How are you doing?

[0:04:37.0] JM: I'm doing great. I'm looking forward to talking about Mapillary and image representation and computer vision. I thought we would start with some basics around computer vision and we can gradually get into the applications of computer vision, the difference between research and applications. Let's start with some fundamentals. If you take an image, an image can be represented numerically as an N-by-N by three matrix of pixels. This is N by N, meaning the dimensions, so maybe it's a 256 by 256 image by 3, meaning RGB, so red, green, blue values so that you can define what the color of any pixel in that space is.

You have these numerical representations of images. Then you can do interesting things with those images. Give an overview of taking an image and its mathematical representation, the matrix and extracting meaning from that image.

[0:05:40.5] PK: Yes. This is actually all that it is that we discuss in computer vision, especially if it's image-based computer vision. If you consider your RGB image that you just described, so it is this N-by-N times three image. What it would really hold is a bunch of numbers, usually somewhere between 0 and 255 per pixel, per color channel. In its combination, as you know this would basically represent the color for each pixel.

This is really just a very structured way of aligning data on a grid, as what we would refer to as an image of – as a human person. Now to the machine, this is basically all it sees. It's just a regular grit, where we have this data arranged. When it comes to computer vision, what you would like to find is and what we would want to do is identify patterns in this image that correspond to something that we as humans would like to identify in this image.

If we take a look at an image and we see that there's a dog, or a person, then to us it's clear that there's a notion of pixels not being there in a random, or shuffled way, but rather they form this object in a very consistent manner, there's a delineation, there's an outline of an object. This is something that we naturally absorb by us just looking at it.

We don't even have to think about it, while as a machine, we need to start building up a representation, or some way of how the machine can actually exploit this structured data that we have and form all of this together and make computations on top of this input, in order to identify something as being an object, something that we might detect or something that we might want to segment.

All of that requires quite a lot of let's say, computational know-how and capacity. If you want to do this in a way that it's actually usable in practice, that you can use it for a business, for applications like self-driving and so on. Yeah, and giving a very short answer on how to let's say, abstract a normal regular input image into something that tells you this is the number of objects that I'm interested in, this is where those objects are located, I think that would fill a number of lectures in a university.

[0:07:58.1] JM: Okay. We'll abbreviate it. We'll get through it quickly, but let's just cover some of the basic concepts. We've got for example an image of a cat and it's represented as a 256 by 256 by 3 matrix of pixels. It's a nice-looking cat. In order to analyze that image, we can do something called convolutions over that matrix of pixels. Can you explain what a convolution is?

[0:08:26.7] PK: Yes. You're basically referring to this current stream of technology, or algorithms that's being used in computer vision. It's all about deep learning these days and deep learning is a way of how can you actually set up extraction of a semantic level from an input image into something that identifies this cat in this image, as an abstraction of layers? Those layers for example, could be convolutions.

A convolution is something it's a local operator and that local operator would locally perform, basically a multiplication and an addition. Depending on what this convolutional kernel, this is usually a 3x3, or a 5x5 kernel that would try to find specific patterns in this image. It would start to react on the local information in that image.

If you think about maybe, think about the eye of that cat, if that would be something like a circular shape, or pattern, then if you have a convolutional kernel that would basically try to have a very similar structure, it would try to find matching patterns in that input image and it would try

– it would be basically firing at this area of where you have this eye of the cat and would be lower in terms of its response on other areas of that image.

As you see, we have a very – in this example, there is a very specific way of having a convolution, or a convolutional kernel that you would apply to an image to find this specific eye of the cat. What you would also want to have is something that might fire on the ears of the cat, or maybe on the body, or on the tail of the cat. As you see, this is basically gradually – it's a gradual process. You would start looking at very, very narrow and local features as we call them. As you basically add more and more of these layers, you would try to explain the whole cat by means of its individual pieces if you want, by assembling this as you basically progress to more and more and deeper layers into a convolutional neural network.

[0:10:34.9] JM: If we were to take a matrix that was a 10 by 10 matrix and it just had ones and zeros in it, you and I would be able to identify, hey look, in the upper right-hand corner of that 10 by 10 matrix, there's a bunch of ones up there. We could agree on that. We could say, “Look, there happens to be a lot of ones up in that right-hand corner.” Then what this convolution system of finding edges or finding circles is doing is very similar. It's just doing that in a deeper, more complex way where a computer can say much like you and I could with our basic analysis could say there's a bunch of ones in the upper right-hand corner, a computer could say this collection of pixels in this space over here, this looks like a circle, or this looks like an eye. Then you can start to build up these higher levels of abstraction that are associated with these matrices of pixels.

[0:11:35.3] PK: Yes. What you would be able to identify is basically those convolutional kernel parameters, they're eventually being learned through this process of training a deep neural network, or a convolutional neural networking in that simple example here. You would actually try to write down your task as a optimization problem. The optimization problem would be solved in a way that you would basically try to find what would an individual convolutional operator have to look like, in order to best possibly explain or solve your optimization problem.

If it's something that would have to respond to the eye of a cat, in the best possible way to solve your problem, because all you want to do is finding eyes of cats, then that convolutional kernel would look probably very much like an eye of a cat if that would be possible. What it would really

try to do is find exactly the pattern that best describes your data that you use for learning this. Basically, you're training data that you have in your learning problem.

[0:12:39.9] JM: Right. Yeah, you might have a collection of images of cats and somebody has gone through those images and labeled the eyes of the cats, so that the computer can identify that this section of the picture is a cat eye. There's a number of other terms we could go through, but I don't want to spend too much time on these basics of computer vision because there's a lot of other podcasts that have covered this. One other term I'd like to define is latent space. Can you explain what latent space is?

[0:13:12.8] PK: If you refer to a latent space in terms of a convolutional network, or deep neural network, then what you've basically started this discussion with was this RGB image, this N-by-N times three matrix that you use as input. At the end, you want something like an output. You want this label that basically says this is a cat. This image shows a cat. Now everything that you use in order to have this translation from this original input data to this let's say, output statement that says that's a cat, this is largely undefined. This is not something that we can as easily understand that this is the input image, or that's the output that comes.

Everything that's basically in between can be in this hidden layers, in this hidden or latent states that you have, that you can have your model operate in. That means you basically undergo a number of different transformations from the input to the output and all of those states are somehow latent, because you don't really define them in advance.

[0:14:15.1] JM: Right. I guess, as the network that you're building gets deeper and deeper, you're getting further and further from these well-defined operators like an edge detection layer, or a circle detection layer and you're getting into a higher and higher level territory where the computer is taking things like edges and circles and some of these lower layers and abstracting them into a latent space. That latent space may cover features that are harder to describe to a human, but nonetheless they are based on patterns. It's just latent to what may be a surface level understanding of these collections of images. Is that an accurate way of looking at it?

[0:15:05.8] PK: Yes. This is also very often referred to as hidden layers or hidden states and that's very much it. Yes. People started to visualize these deep neural networks and because

they want you to understand what's basically under – what's going on under the hood. As you just said, so very early on in these layers, you very often find responses for edges, for very sharp transitions, so really very, very low-level features as we would call them. As you basically go down deeper in the network layers, what you would find is something that agglomerates this local information and is getting more and more – it's starting to assemble it and become more and more a response that looks like the actual object you would want to find.

[0:15:50.6] JM: I think we've covered the basics of this, but what gets people so excited about deep learning is this idea that you take something like an image. We look at image as more like, how on earth would we explain to a computer what an image is, or a music, a piece of music? You have this highly dimensional abstract concept like music, or a piece of art, but then we've just defined this way of breaking it down into mathematical terms, into a structure that a computer can understand at a lower level and then we've built up these richer abstractions that let us define things that maybe we can't even put into words. We can just label them and describe into a computer and the computer can learn the mathematical space and then the latent space that composes those concepts. That's what's so exciting about deep learning, at least to me.

[0:16:44.4] PK: Yes. You're absolutely right. Maybe to just fill in this hole of what happened before, let's say 2010, 2012 when this started to really fly in this computer vision space, so all these deep learning and narrow artificial intelligence methods that we're using today. Before, let's say end of the 90s, beginning of the 2000s, people were working on very different models and methods and they wanted to be much more explicit in the way they wanted to solve computer vision tasks.

There was an era where people were very much into describing objects by in terms of their shapes and trying to find shape models that would understand, or well explain the objects you would be interested in recognizing. While as you say with deep learning, now all of a sudden what you would really need is a way to first of all, have a curated way of collecting data. For example, you have large data sources. Those data sources are somehow being annotated, can be most of the time we're talking about human annotations, so there's really a person sitting down and annotating this data. That's one very important part.

If you have large masses of those annotated datasets, this will already help you to tackle your problem from one perspective. The other perspective is you have a model that as you say, you basically define what is the input and what you would like to get as an output. Now and let's say in a simple example of performing let's say, image classification, where you don't really care about the very fine details, but you would just have this N times, or N times N times free input image. Then you would want to have a single label attached to this in the outside. Or sorry, as a result from the model.

The second part also needs the computation that you need in order to train these models, but as you say, the really exciting part is definitely that you define this problem in terms of its mathematics, and then you start basically solving this underlying optimization problem. If you just train it long enough and if you have good enough data that explains your problem, then the model will eventually find a way to absorb this information in your training data and come up with something that if you basically show a new and unseen image to this model, it will still give you the correct answer for what you would want the model to predict on this data.

The expert engineers at Algorithmia are always available to help your team successfully deploy your models to production with the AI Layer tooling. You can also listen to a couple episodes I've done with the CEO of Algorithmia, if you want to hear more about their pretty sweet set of software platforms. Go to algorithmia.com and use the code SWE Daily to try it out, or of course, listen to those episodes.

[INTERVIEW CONTINUED]

[0:21:03.4] JM: There's been an evolution of these architectures, of these convolutional neural network architectures. There's this AlexNet, VGGNet, GoogleNet. I don't know much about the history of these things. Could you give a brief history of how the CNN technology has improved through this research evolution?

[0:21:26.1] PK: Yeah, I can try. I mean, those CNNs, those convolutional neural networks, they're actually not new. I mean, people like [inaudible 0:21:33.3] who's one of the founding fathers, or Geoff Hinton, they were working on this fairly a long time ago. Actually CNNs came up and of the 80s, beginning of the 90s and they started the fairly shallow networks by that time. They have a very small, very low capacity, as we would say only a few layers and only very few parameters.

Basically, you would parameterize these models, you would say that's the number of neurons per layer that you would want to consider. Those were really small at the beginning. Then let's make a large jump in terms of time. At some point as you said, there was this AlexNet. AlexNet was I remember well. This was a paper presented at NIPSP in 2012, which is one of the largest and most important, or actually now one of the largest and most important conferences in the field.

They were going down, I think it was something like seven, or eight layers with a large number of parameters. Don't name me down on the actual number of layers, but they just increased the capacity a lot. I was thinking, “Wow, that's really insane. How can you go that deep?” Then we made another jump and make another jump in time. All of a sudden, there's this V2T networks and later on, what came in here and collaborators found the virtus residual networks, where you

all of a sudden go down 50, 100, 150 layers, even more and there's a large number of variations that came out afterwards. All of a sudden, you go down really extremely deep.

For a long time for specific tasks like let's say, image classification again, you could see that adding more capacity in terms of adding more and more layers to the model, I mean, it came with some complications on how you could properly train those models, but if you had the data and the compute power and the time to actually wait for those models to be properly trained, then they would just start to perform better and better the deeper you went.

Yeah, and there's still a trend of adding new parameters if you have even more data sources, or even larger data sources, then there's still some increase in performance that you see. Of course, this always comes with a cost, the cost of the computational budget that you can spend.

[0:23:52.8] JM: Yes. This sets us up well for a discussion of Mapillary, because as you have defined, this application of deep neural networks to computer vision is fairly new and it's still moving. It's still getting more efficient, it's getting more accurate. That's the tailwind that you're riding at Mapillary. That's the answer to the why now question, why does Mapillary make sense now, when some of the technology that Mapillary is built on has been around for five or 10 years, but other pieces are very, very new? That's the context I'd like to frame your business in. Can you explain what Mapillary is?

[0:24:40.6] PK: Yes. Basically, Mapillary is a platform where we collaboratively collect street-level data and we take it from any contributor, any type of camera model and from anywhere in the world. Basically, this is a very, very large platform that we're operating in order to build a joint representation of the world.

[0:25:05.4] JM: Can you give more detail on how you're building that joint representation of the world at a high level?

[0:25:10.9] PK: Yes. We have two very large building blocks. Leaving all of the backend aside, which I think you greatly covered in a discussion with Peter Neubauer, our CTO some time ago. We have the two main computer vision pillars we build on is 3D modeling and object recognition, or in particular, semantic segmentation.

Maybe we should focus on semantic segmentation for now. This is basically – so before we were discussing about image classification, you give an image and the machine tells you what's basically the most likely class, or object category it is showing. For semantic segmentation, the task is a bit different. The task there is given an input image, you would assign as categorical or semantic label to every single pixel in that image. For us, this means we are for example interested in a very large number of street level assets and objects, or points of interest you might want to call them.

Assume you have a dash cam and this dash cam is on and you're driving towards, say a traffic light and you have to stop because it's red. It will cover a large field of view from your dash cam. You would see maybe a crosswalk, you would see that the traffic light, you would see other cars, you would see a very large number of objects in those images. The machine and when you basically run semantic segmentation on this image, would try to recognize every pixel of every object that we're interested in.

Indeed, this is something that Mapillary uses a lot for a number of its services. This starts with, let's say privacy protection, where we automatically identify the areas where you see human faces and blur them or see license plates and we blur them as well. We used that to extract map information, so we actually identify let's say, this traffic light again, we would find all the pixels in this image that would form this traffic light. If the second pillar that I mentioned before with this 3D modeling, we would find a way to actually geo-position this traffic light in our model of the world, in our freebie model of the world.

For in the meantime, we have a 152 different object categories that we were able to recognize. We would start pulling out all of these map objects individually and we will position them on a map and make this data available. That's what Mapillary is all about.

[0:27:45.3] JM: Okay. Let me see if I can explain it accurately. You can tell me where I'm wrong. You've got people contributing images from their cellphones. You've got just people taking pictures with their smartphones. You're crowdsourcing 2D images of the world, so you're getting this large influx of 2D images that people are taking from their phone. When you take a picture

from your phone, it can automatically geotag it, like where is your latitude and longitude in the world, which is not perfect, but you can get it to some degree of accuracy.

If you imagine just people taking random pictures from all around the world and they're somewhat geotagged, maybe you can derive what angle they're taking the picture from, then you can take these 2D images, you can synthesize them together to construct a 3D scene. That's one of the things that you're focused on is taking all these 2D images and synthesizing them into a 3D representation of the world.

Then another application that you just mentioned is the process of looking at those images, those I think you can look at them just as the 2D images, but you can probably infer additional information from the 3D representation. You look at the 2D images, you use object segmentation which is another area of computer vision that's rapidly evolving. We did a show a while ago about object segmentation, where you can define what are the prominent shapes in this 2D image.

There's a square object here, let's draw a bounding box around that and segment it out and then maybe we can do some further processing on it, now that we've defined that there is an object here. You have this object segmentation thing that's also running. Then once you get the object segmented out, you can do classification on those images. Maybe if you've got a big scene of a 2D image of a street and you run object segmentation across all the different objects in that scene, then you see a stop sign, you see a Prius, you see – well, the computer would just see here's an object A, here's an object B, here's an object C. Then you could pass those bounding boxes to a further classification algorithm.

You say, "Okay, what is this bounding box here? What information is in this bounding box?" The computer might say, "Oh, it's a stop sign." Then you know there's a stop sign in this image. Then you can start to work with using the segmentation. You can derive what's going on in the scene and then you're building a 3D model of literally what exists in the world. You now have a 3D model that you can explore together with a way of communicating to the computer, this is a stop sign in this 3D world.

Then you could ascribe – later on, or maybe you're already doing this, you could ascribe meaning to what a stop sign is, so that a computer could eventually understand, here is a 3D model of the world with this red object in it, that it happens to be a stop sign. What a stop sign is is it typically tells people to stop if they're driving. Now tell me what I got wrong in that representation of Mapillary?

[0:31:03.1] PK: The basic building blocks are all there. You really named them all. The first one is the 3D model itself. This is really, you have this sequence of images that you would capture with our app, if you download it, or if you use your own rig, your own capturing software. If you have this together with the GPS signal, you would take those images, you would upload them to our servers and we would run something that's called structure from motion.

The structure from motion is something that allows you to basically extract the 3D, three-dimensional structure, a three-dimensional model based on a sequence, or at least multiple two-dimensional images by basically finding correspondences in those images. You would re-identify the same points and then it would allow you to infer a 3D model of the world.

That's already something that's very important, because this would allow us to tie together this information that we have in a redundant way as we capture, as we for example drive with our car.
The second part where we basically take a detour to what you described before is, so we're running semantic segmentation directly on the raw input images. The semantic segmentation as I tried to explain before is it would really label and assign this semantic label to every single pixel in that image, which is a large number of pixels on this – even on this small tool 256 squared example that you mentioned before.

It would already try to locally find consistent segmentations consistent with what we see as objects on this image, as we interpret from a human perspective. Basically, this dis bunch of pixels all belonging for example to the same car, or to the same traffic light, we would find a way to group those pixels and then together with the 3D model, because this actually came from the same data, we would know what the correspondence is between the position of those pixels in the 3D model that we extracted before, together with this semantic interpretation from our semantic segmentation algorithms.

Obviously, there is some grouping step involved, which I think you were referring to before as a detection mechanism. What we would run at our platform is actually some clustering algorithm. Clustering together those basically pixels that contribute to this object as a traffic light again, or as a surveillance camera, or whatever we were finding, this would help us to really geo-position this on the map. The whole pipeline is really give us a sequence of images and we would try to extract all of the relevant street-level assets and data and points of interest that would help you if you have an application as a self-driving car, to maybe find your way around a certain place.

[0:33:58.8] JM: This problem that you mentioned there, structure from motion. Explain in more detail what structure from motion is and how you're applying it.

[0:34:10.3] PK: Structure from motion as I said before, it's a way to infer some three-dimensional structure from two-dimensional images. Basically, if you have two images and you take them in a subsequent way, so if you basically just – if you walk along a certain path and you would take two images subsequently, you would very often find similar things in those two images, right? Or actually the same thing, just shifted by some motion.

If you would actually be able to identify the corresponding parts or features in those images, something like corners, or other prominent points that you would find with some feature detector, then you would be able to infer the 3D structure based on those correspondences that you would find.

[0:34:58.6] JM: Why is this important to Mapillary? My understanding of Mapillary is today, most of your image data comes from people with a smartphone taking just a static image with their camera. Why is this relevant to Mapillary?

[0:35:13.7] PK: We somehow need to find a 3D model, right, that we can operate in. This 3D model that we want to identify corresponds to – it's basically a model that we infer from the real world. What we want to do is we want to build a true representation, or something like a scaled model of what we have in the real world, because we want to position, we want to geo-position all those map objects that we're extracting from a semantic perspective.

Even if we were able to perfectly segment each image, if we are not able to basically assign latitude and longitude in terms of a geo-position to this identified object, then this would only be half the wait, right? What we really want to have in our let's say, the most optimal way would be if we had images from all over the world, right? If all those images would somehow show transitions from one location on one spot on the world to any other one, then you would find a path through all those images.

The 3D modeling part, the structure from motion part if it would work perfectly, then this would allow you to basically extract one coherent, consistent 3D model from all of these images. If we then on top can run all of these object recognition tasks that we do, including semantic segmentation and something that will come soon is also recognizing every single instance of a certain semantic class, then this will help us to really extract all of the street-level assets that we care about and all of our customers care about in an automated way.

[0:36:51.8] JM: Right. Whether those images are coming from people with smartphones that are just taking pictures, or people with smartphones that are taking videos, or surveillance cameras that are gathering video, or drones that are gathering video, or self-driving cars with some module on top of it that's recording lots of video, if you can solve this structure from motion problem, you can take advantage of all kinds of different recorded image data and use it to improve Mapillary's view of the world.

[0:37:25.9] PK: Yes, that's right. Obviously, the quality of the data largely depends on the camera, type of the camera and very often, even though those – the cameras that we have in smartphones today are really extremely good already, there's of course still room for improvement. Depending on the image quality, we're also talking here about several types of distortions, so all things that you would in a perfect world you would never encounter, but since we have basically data from any camera model that you can find out there, all of that contributes to the accuracy and to the quality of the end-result.

For example, if you go back to this structure from motion, or to this 3D modeling problem, what you would have, so once you have those features that you would like to re-identify in matching images, there's this reconstruction part where you try to solve and find basically that the camera

pose and the 3D points and the two cameras that you find and all of these things largely depend also on how well all of these data matches.

Eventually what we try to build and that's something where we try to invest more let's say, brain power is to come up with models that can deal better with these, say imperfect conditions, and that we have by accepting any kind of, let's say source of imagery. Indeed, we find of course, there is some difference if you have a let's say, super expensive capturing rank that's being used to capture images that's at a very high quality, as opposed to taking a very let's say, an older generation of a smartphone, which has maybe issues with the shutter of the camera, or other types of problems. Trying to align all of these let's say, challenges is what we're trying to address in our works.

[0:39:24.4] JM: Well, what makes it an exciting company, at least from an external perspective and I'm guessing from somebody who's on the inside is the problems that you are faced with at this point, to getting to Mapillary's vision of having this really richly defined 3D vision of the world that people could query and they could build other applications on top of, build self-driving cars, build drones, build all kinds of things on top of a structured labeled 3D model of the world, like that's really exciting.

The barriers to you implementing that full dream product set, the problems are mostly incremental. As far as I can tell, it's like engineering problems, it's the engineering problems you want to have. It's like, what is our data pipeline look like? Do we have enough storage space here? Do we have enough memory here? Are we batching these models correctly? Are we training these models correctly? Are costs under control? Is the cloud computing infrastructure budget that we've allocated, is that enough money to actually afford all this computer vision processing? Are there any real big scientific bottlenecks that you're worried about, or do you feel you're at the forefront of something really exciting and it's just a question of implementation?

[0:40:47.8] PK: I think, this is a really interesting question, because we're always trying – so my team and I, we're trying to work on the scientific forefront definitely. For example, one problem that we spend a lot of time on working was recognition problems in general and the semantic segmentation in particular. If you went back a couple of years ago, those models were just

about to start working and there's a number of let's say, benchmarks and datasets, where as a scientist you start developing your model, you implement it, you fail a couple of times.

If you're lucky and persistent enough, at some point it starts out working. Ideally, it starts working really well and then you go take your model and put it on those publicly available benchmarks. If you see that you make basically a significant margin on what is out there, then there's of course always some additional scientific value that you can add on top of this.

Is it on how to get there in a more efficient way, in a more computationally efficient way, or also in terms of runtime, in terms of let's say, that the final output metric that you care about, all those parameters there's usually always room for improvement.

The second part that was in your question was is it just an engineering perspective? I think there's a large aspect of engineering indeed, especially having all of these things run it scale, run it at the budget that you can actually afford and at the same time, scale that system in a way that you can basically handle any amount of data, let's say any under-under quotes, a very large amount of data and make it all work together, so this is a really fantastic engineering piece that mainly our back-end team and our computer vision team are solving.

While for example, myself and I we have more the privilege to really work on the research direction where we don't, or at least not – the first consideration is can we bring that to the very large scale. To us, it's more can we explore a new field? Some new, let's say inside for an existing problem, or can be – and would be even better of course from a research perspective, come up with something that hasn't been done before from an application perspective and this is how this whole field evolves and how so many companies and universities and research labs are contributing to making all of this knowledge available. Then for a greater benefit, I think that's a really exciting part and I'm very happy and lucky to sit right in this field.

You can find out more about Mesosphere's Kubernetes as a service by going to softwareengineeringdaily.com/mesosphere. Mesosphere's Kubernetes as a service heals itself when it detects a problem with the state of the cluster, so you don't have to worry about your cluster going down. They make it easy to install monitoring and logging and other tooling alongside your Kubernetes cluster.

With one-click install, there's additional tooling like Prometheus, Linkerd, Jenkins and any of the services in the service catalog. Mesosphere is built to make multi-cloud, hybrid cloud and edge computing easier. To find out how Mesosphere's Kubernetes as a service can help you easily deploy Kubernetes, you can check out softwareengineeringdaily.com/mesosphere, and it would support Software Engineering Daily as well.

One reason I am a big fan of Mesosphere is that one of the founders, Ben Hindman is one of the first people I interviewed about software engineering back when I was a host on Software Engineering Radio. He was so good and so generous with his explanations of various distributed systems concepts. This was back four, or five years ago when some of the applied distributed systems material was a little more scant in the marketplace. It was harder to find information about distributed systems in production, and he was one of the people that was evangelizing and talking about it and obviously building it in Apache Mesos.

I'm really happy to have Mesosphere as a sponsor. If you want to check out Mesosphere and support Software Engineering Daily, go to softwareengineeringdaily.com/mesosphere.

[INTERVIEW CONTINUED]

[0:46:00.8] JM: We're talking about a company here, but you have a research division. Mapillary has a research division. You're leading that research division. Where does research end and engineering implementation begin? What's the division of labor? How do discoveries in your research department make their way into the implementation department?

[0:46:23.8] PK: Yeah. One of the very nice things to have in a startup environment is, so ways are usually short, right? We're still below 50 people working in the company, especially on the engineering and research side, it's much less. If we find something that actually turns out to

work on these benchmark data sets that I was mentioning before, including for example our very own Mapillary vistas data set, which we appreciate a lot, then we will definitely have the attention from our let's say, more product and implementation and scale-oriented team that will take this research and will want to bring this into production as soon as possible.

The very nice finding, again this is in a startup setting, this works like super in a very, in a super smooth way in my opinion. Indeed, what we found and what we're very proud of are our very own semantic segmentation, let's say developments and advancements, which also helped us win a couple of public benchmarks this year. All of this technology has gone into production, I think within a couple of months. Yes.

[0:47:38.0] JM: Wow, that's pretty quick. You have this research department. The back and forth between industry and academic research and development, I mean, this has been a long story in technology, like what is the responsibility of the academy? What is the responsibility of the industry? Or what's the not responsibility, what's the relationship like? What's the push and pull and what's the value of these cross-cultural areas like the – it's like NIPS. NIPS has this place where people can cross-pollinate.

What are you seeing in the cross-pollination between the academic world, or the "research world" and the implementation world more broadly? How much information are people willing to share? What are they willing to share? Is it more open? Is it more closed off? Where is the proprietary secret sauce? Give me a bit of framing of that culture today.

[0:48:42.7] PK: I think there has been some shift. Again, looking back from let's say, 10, 15 years to now, I think what used to be considered as purely academic research is something that maybe has arrived more and more at the company research level. For example, look at all those large Silicon Valley companies. I mean, they all run their research labs. Most of them actually publish, disseminate their results and make them publicly available, not only as a form of let's say, a technical report or a paper, but also in terms of code; code that can be used on the very friendly license, so also under commercial settings.

I think the reason why this is done is many-fold. One of them is probably – so there's a large number of people that would maybe just like to stay in academia, because they like the scientific

way of how they can start to develop new things, how they can experiment fail, succeed and undergo all of these phases that you have as a researcher.

At some point, if you find something, you write it up, you submit it to conference and hopefully it gets accepted as a paper. Those people have been – as the field has been growing and as there has been more and more demand also from an application point of view, and I mean, this is largely due to deep learning that those things actually started to work and used to become more interesting also from a commercial perspective. There was a much higher demand all of a sudden for people with those skills.

If you wanted to attract those people that would actually have preferred to sit more in their academic setting, if you offer them let's say, a great workplace in terms of have them still work on their problems and the outcome that they get is actually can directly be translated into a product, or into some commercialization. If you in addition don't lack resources for let's say, a compute infrastructure, or data, or any other ingredient that is necessary, I think that has contributed a lot to make this whole community more open in terms of what is being shared, but also in terms of what can be used for anyone and made available to anyone.

[0:51:05.7] JM: You also make money when you do that. The people who – the vast quantity of people who are going to industry from the purely academic world, the more that that happens, the more it makes certain academics become really gems, because they become these nonpartisan forces. You have people like Lex Friedman at MIT, who had started a podcast recently that's really good actually, MIT AGI or something. It's about artificial intelligence. I recommend anybody that likes this podcast to check it out.

He's nonpartisan. There is one thing I've been enlightened by since starting the podcast. At the beginning of the podcast it was like, why is there still academic computer science? Why hasn't this all moved into industry? In the industry, you've got unlimited compute, you make more money, you have these things that can be put into products. Academia is politicized in its own way, but it's also removed from politics in certain ways, because there's politics that you just get pulled into by virtue of being at a specific company that you are immune to in academia.

[0:52:18.2] PK: Yeah. Maybe, I would probably say that there is – every field has its own rules, by which you somehow need – you need to understand and you should understand, if you want to participate in a certain area. Maybe you coming back to your other question, I think there's still a large number of open and unsolved problems. These deep learning is something, if we talk about this, if we still consider this mostly from this supervised setting, right? There's this data that has been annotated by someone and then there's this machine that tries to find the optimal solution given this data set and trains this model and spits out something on new data that matches very well ideally what it has learned on the training data.

We're talking here about supervised learning. That's a very narrow space of machine learning. If you think about how much more data that is out there that hasn't been labeled because it's simply not feasible to label all the data in the world, I mean, there's virtually a much, much larger area that should receive scientific treatment, that people should come up with smarter models how to deal with those masses of data. There's still enough work out there.

[0:53:36.9] JM: Right. Now, I know we're running out of time here. There's a couple of macro-questions I have for you. I read this book recently, *AI Superpowers*, Kai-Fu Lee's book, which is really good and it's about the implementation of artificial intelligence technologies, mostly in China, at Chinese startups and how the speed to deployment is faster, in some ways more efficient, in some ways better quality bringing these models to market than places in the West. Part of that is due to they have richer datasets for a variety of reasons.

Have you seen a difference in nationality across who was good at research, versus implementation in computer vision? Specifically, have you seen any ways in which China stands out as being particularly good at computer vision, or just particularly good at machine learning, or is this more of a circumstantial thing?

[0:54:39.1] PK: I think I couldn't make a very general answer on this. My personal observation is that so while attendance to these conferences like NIPS or CVPR is growing at a very large pace, I think there's a much larger crowd now maybe from Asian countries these days. I think there's a really large, let's say cosmos of startups that's starting to build up there. As you say, a number of young companies that come from, for example, from China, they have shown outstanding results on many benchmark datasets.

Having those contributors, I think it's a matter of many things; engineering is very important. Having the curiosity and the let's say, the drive to find new ways to find better ways of designing models, of training them. There's of course also a large financial let's say, backup for many of these companies. They can afford to buy large cluster farms to try out a lot of things. I think this is a very multifaceted answer on this.

Again, in the in the particular cases, I'm sure that you can find let's say, those very outstanding researchers and engineers all over the world. What is definitely true is that there's a strong rise, especially from people from China and from companies in China these days.

[0:56:08.8] JM: Yeah. I'm looking forward to hearing more about this area, because computer science research has been this world of cross-pollination and information-sharing and positive some information sharing. It seems like with computer vision, you have this opportunity of almost unimaginable value gain, value creation that could be made that is not zero-sum. It's not like if China gets really good at computer vision, that takes away from us reaping the rewards of really compute, of really good computer vision.

The value creation that could exist is just so unimaginably large and it makes me optimistic that maybe we could have some geopolitical cross-pollination, at least, at least between scientists. To some degree, I think we're seeing that. Again, I'm not an expert on this. I hope to learn more and hopefully interview some people in the future, if anybody out there listening has suggestions for people to interview.

Okay, so we're at the end of the time. Last question, I'm a little less optimistic about the spamming of our internet with fake images and fake videos, particularly the fake videos because somebody I talked to recently said that we really don't have good mechanisms for detecting these deep fakes. Now, the deep fakes are obviously not very high-quality yet. I've seen some naive solutions to improving deep fakes detection, like how rapidly the fake video of Trump is blinking. It's not accurate with the human, so you can detect that this fake Trump video is fake by the blink rate detection, which is interesting.

We're entering this world where we're going to have spam versus anti-spam on the front of what is real video. Do you have any thoughts on this area, or are you as concerned as I am, or are you less concerned?

[0:58:08.8] PK: I think this is a very interesting question. From a scientific perspective, I would say that there's probably as many works that try to reveal, or identify let's say, fake videos, or videos that were artificially generated. Some of them might be let's say, more successful than others, but I think with every paper that comes out to generate this fake material, there's for sure at least one other paper that would try to identify if that was a fake video or not.

In the way this is actually being used, that's definitely a completely different story. I'm sure that you can – that there are ways to use this manipulated data on very, very mean ends and for very let's say, reasons that are not forming a greater benefit for the society. We should as a society be in general, be more aware of what's going on. That you should at least think twice before you start just taking something and just because you see it as a video, take it for granted.

This is an education process that I think also needs to start and happen, such that you maybe start learning more of what is possible from a technological perspective, but also be more aware from how we take information and how we deal with that.

[0:59:33.0] JM: All right. Well, thanks for humoring me on something that's totally unrelated to Mapillary. It's been a real pleasure learning about computer vision from talking to you. I'm sure we can do more material in the future. I know we didn't – we probably didn't dive into Mapillary as much as we could have, but I wanted to just get your take on a bunch of different things. Thanks for coming on the show, Peter. Been great talking.

[0:59:55.3] PK: Thank you very much for the invitation.

[END OF INTERVIEW]

[1:00:00.3] JM: Continuous delivery frees up your developers to ship code independently of one another. GoCD is an open source continuous delivery tool. We've partnered with GoCD to deliver some quick tips about modern continuous delivery.

Today's tip is to visualize your workflow. You always want to be able to see the status of builds across your pipelines. Visualization can help you step back and see the bigger picture of your continuous delivery process. Your continuous delivery tool should help you visualize your workflow. I've found that showing the workflow visualization to new hires that are unfamiliar with continuous delivery can also be quite useful for explaining how your release process works.

To find out more about continuous delivery, check out gocd.org/sedaily. GoCD has been a sponsor of Software Engineering Daily for more than two years and we've done lots of shows with the team from GoCD. It's a great team. We're proud to have their support. If you're getting started with continuous delivery yourself, check out gocd.org.sedaily.

[END]