**Interviewer 1:** [00:00:00] Could you confirm it on records that you agreed that it would be recorded?

**Interviewee:** [00:00:04] Yes. I agree to be recorded.

**Interviewer 2:** [00:00:06] Thank you.

**Interviewer 1:** [00:00:10] Thank you. Thanks a lot. So I would just start with some questions about your background. What is your current position at your job?

**Interviewee:** [00:00:21] So my background and my current position in my job, right?

**Interviewer 1:** [00:00:23] Yes.

**Interviewee:** [00:00:24] So my background is that I was born in [Country: Removed for Anonymity], but I moved to [Country: Removed for Anonymity] on 2012 to do a masters at [University: Removed for Anonymity]. And then I did my PhD in [University: Removed for Anonymity] in [City: Removed for Anonymity], there under the [Project Name: Removed for Anonymity] project, which is a Marie Curie Fellowship.

**Interviewer 1:** [00:00:41] Okay.

**Interviewee:** [00:00:42] And then in 2017 I came back to [Country: Removed for Anonymity] to the [Research Center Name: Removed for Anonymity] of Artificial Intelligence, where I'm currently also a like a postdoc and researcher in general.

**Interviewer 1:** [00:00:53] Okay. So you also work as a postdoctoral researcher right now, right?

**Interviewee:** [00:00:58] Yes, I got my PhD like three weeks ago, like officially got my diploma three weeks ago.

**Interviewer 2:** [00:01:02] Congratulations!

**Interviewer 1:** [00:01:05] Congratulations. I got mine just in January. So...

**Interviewee:** [00:01:11] Freshly minted PhDs.

**Interviewer 1:** [00:01:17] Yes! Okay, so could you tell us about your overall work experience and experience in deep learning systems in terms of years?

**Interviewee:** [00:01:31] Well, I have, when I was working in [Country: Removed for Anonymity] I had like four years of professional experience, mostly developing software like in Java and Python, and then more scientific software will be like three or four years like during my PhD and now during my postdoc.

**Interviewer 1:** [00:01:47] Okay, so your, the experience during your PhD doesn't count towards deep learning systems?

**Interviewee:** [00:01:54] No no, it is. I started working, using Keras actually during my PhD. It was extremely helpful to use, to have Keras available. Yes.

**Interviewer 1:** [00:02:04] Okay, thank you. So could you tell us what type of deep learning networks you have developed/implemented and by that I mean like supervised/unsupervised/reinforcement learning and etc.

**Interviewee:** [00:02:16] We worked on supervised and unsupervised learning, usually with convolutional and with recurrent neural networks.

**Interviewer 1:** [00:02:24] Okay, thank you. And which problems were you trying to tackle using these deep learning networks, like an example could be image classification, speech recognition and etc.

**Interviewee:** [00:02:36] Yes yes, it's mostly image classification, object detection and pose estimation. In general, we have also done some things that are more like outside of that, like prediction of parking occupancy, which is a regression problem. That's like no images, using time series data. And, yep, that is I guess.

**Interviewer 1:** [00:02:54] Okay. Okay. Thank you. And you mentioned Keras, but did you, which programming languages and frameworks have you been using in your experience?

**Interviewee:** [00:03:05] Well at the beginning I used Lasagna(?), you probably have heard of it, using with Theano, and then I started using Keras with Tensorflow, when it appeared in 2015 or 2017.

**Interviewer 1:** [00:03:18] Okay, so I would guess that as a programming language you were using Python.

**Interviewee:** [00:03:23] Yes. Yes, mostly Python, yes.

**Interviewer 1:** [00:03:25] Okay. So for this interview, we have one general question, which is what types of bugs/problems/challenges have you faced when developing these systems. And we're interested in all kinds of bugs and problems, starting like conceptual bugs, general programming bugs, any crashes, errors, everything. If there is anything on your mind about that, you could start from there.

**Interviewee:** [00:03:51] I mean Keras has a lot of bugs, I mean it has improved over time, but in general, for example, I remember during my PhD that, I think in 2016, in 2017 Keras 2 appeared, there was Keras 1 and then they changed the API and some things changed. So models start you trained before didn't work anymore. But there was of course no error message, just the predictions were wrong. And that's of course very annoying.

**Interviewer 1:** [00:04:22] Okay. Yeah.

**Interviewee:** [00:04:24] What I did during my PhD, I decided to freeze the Keras version I was using, so not to have these problems while I was writing my thesis. And I think this happened because they change the implementation of batch normalization and all the models are by using batch normalization. So yeah, I mean it was sort of something super weird. And some people reported it as a bug in the GitHub repo. But in the end it was not really well understood why it happened. And it was combined between some issues in Tensorflow, because in Tensorflow they were trying to kind of standardize the API, to be more like NumPy-like and they changed about normalization implementation several times. So if you train a model one implementation, it doesn't work with other implementation, but there are no error messages. So probably the most kind of annoying bugs is the one that you get the wrong output, but it has no error message or anything that tells you that is wrong, that you have to look at the numbers and they are wrong. Yeah, I think I remember that the problem was that my models were overfitting on the same dataset that I trained them, like in the training set they were overfitting very heavily, so that made no sense. Because during training you get zero error, but during testing you get 100% error, and then I re-trained the modelo then it worked. Okay, that shouldn't happen.

**Interviewer 1:** [00:05:46] Okay, anything else?

**Interviewee:** [00:05:49] Well, yes, there are a lot of issues, actually. Another issue is that it is very hard to make like custom layers or custom loss functions. I mean if you read a paper and then you find implementations, usually using older frameworks, like using Caffe or something else, and then we had to translate this to Keras and it's very difficult to make the loss and very difficult to debug it, because the whole system is kind of symbolic. So you cannot just run the loss function with some importance amount, because then Keras calls it like in a batch of predictions and it's kind of a black box. So I think there's no like implementation inside Keras to actually develop these things easy. So I was also working with students doing object detection and they were implementing, for example, SSD - the single shot multiple selector, and we had a lot of issues that we had to kind of make a working environment just to debug the issues with the loss function and with the custom layers that we were implementing. And that, in general, is I think the biggest issue we had with Keras.

**Interviewer 1:** [00:06:49] Okay. So, so far you were telling us about the bugs in the framework itself. Do you remember any kind of bugs that were there because of your let's say wrong implementation and so on.

**Interviewee:** [00:07:02] Yeah. Yes. That always happens. I mean one thing is understanding what the scientist in the paper wrote. Another thing, is that, maybe, the scientists didn't understand what they were writing and they made a small mistake, for example, in what the equations are. And then you have to maybe, go to Reddit or go to the internet to find the action implementation. Yes. Okay. That happens. And what also happens is that some algorithms are very simple, but it's very hard to implement them using, I think, using the tensor concept. For example,Team SBP(?), it's a part of Faster CNN, that allows that you can have a neural network that takes inputs with multiple size, with variable size inputs. So I give you a fixed size vector, but it takes variable size inputs and then inside there are no implementation of, that are kind of reliable. I mean, Tensorflow has an implementation, but it only takes fixed size images. So it kind of defeats the purpose of using this and there are some limitations on GitHub, but a lot of people say that it doesn't work, doesn't compel anymore. Because usually someone developed a solution, for some specific version of Tensorflow, and then maybe the move on, and the implementations in GitHub, it doesn't work anymore, needs updates, because the API of Tensorflow changes and then no one knows how to make it work, and then you have the code, but it doesn't work anymore. I think that's one of the biggest issue about kind of framework level or conceptual level.

**Interviewer 1:** [00:08:41] Okay. Thank you. I was asking if you have anything else to add?

**Interviewer 2:** [00:08:50] We did not hear you.

**Interviewee:** [00:08:55] Ok, any kind of new bugs. Um, well if you ask me about what I have seen in StackOverflow is that in general people they don't have the background to use Keras or the background to use Tensorflow. They are extremely new to linear algebra and to neural networks, and that's usually the biggest, I think the biggest problem that if you look at the statistic of questions, I think the biggest question is that they get this error that says that the dimensions of the layers, they don't match. Like Keras says, I expect this as the shape, but this is the shape that you provide. That's the most common problem. People don't understand why it happens. And it's very easy, because it's just that if you read the wrong information to Keras, then of course it would give error at some point.

**Interviewer 1:** [00:09:35] Okay.

**Interviewee:** [00:09:36] Yeah, it's more like a conceptual thing.

**Interviewer 1:** [00:09:39] What is the second most popular questions that they ask, if you remember?

**Interviewee:** [00:09:45] Well, in StackOverflow, there is also a lot of issues that ask about algorithmic or not-algorithmic, but like model, like they take a model and, for example, for the MNIST dataset or for CIFAR10. And they try to train on these images and it works. What doesn't work, as they expect, like they say "Oh, yeah, I got 99 percent precision and accuracy in MNIST". And then they try it on face images and then you get 10% accuracy. So yeah, and then people ask like, how can I improve the model. The first issue is that that's not a programming question, so usually it is offtopic for StackOverflow. And, in general, it's not like, you need to do scientific research to actually find the right model, right? There are papers that are about like networks for face classification, networks for object detection. So that one model works for one kind of data, doesn't mean that you will work at all or well in another kind of data. And usually people overlook that a lot. Like you to take a toy model and they apply it to a real world problem, and of course, it doesn't work.

**Interviewer 1:** [00:10:45] Okay. So in your experience the problems that people ask on StackOverflow and the problems that developers face in real world systems, are they like very different or just...

**Interviewee:** [00:11:00] Yes, they are two worlds, completely different worlds. Yes.

**Interviewer 1:** [00:11:03] Okay.

**Interviewee:** [00:11:05] Okay. Yes, and I think another group of kind of big group of issues they come out when Keras updates. Because the things that in the last eight months, I think, because we are in July, the last release of Keras came on December of last year, I think. And there have been a lot of changes in GitHub, at the documentation that is in the website. It fits directly from GitHub, but there's no release of Keras since like in the last eight months. ,So sometimes people ask, for example, the documentation says that I can use this function with these parameters. But it doesn't work in the latest Keras, and that's because they have to use the version from GitHub, the master, until of course, they are completely, they don't know that. And people ask, and no-one knows what the problem is, because no one is using this feature. And usually I look at the documentation, like to see something has changed. So that's also like documentation issues, because it is not in sync with the stable version.

**Interviewer 1:** [00:12:03] Okay. Okay. Thank you. Thanks a lot. So I wanted to ask, so in your experience, did you usually use existing datasets to train your network or did you ever had to collect your own training data?

**Interviewee:** [00:12:19] I had to collect data for my PhD, because we were doing like object detection in sonar images and there are no public datasets about sonar images. So we had to collect, and capture, and label data.

**Interviewer 1:** [00:12:32] Okay, so could you tell us about the problems and bugs and issues that you face when collecting training data?

**Interviewee:** [00:12:41] But you mean like from the Keras point of view or from my point of view?

**Interviewer 1:** [00:12:47] Overall, from your point of view, yes.

**Interviewee:** [00:12:50] It is difficult, because one thing is, okay you need to collect data. So in our case, we had to use the sonar sensor, put some objects in a sonar environment and then move it around to capture some data. And this is very error-prone, like first we need to move the objects, you need to introduce variability. Because most of the public data sets to use color images, so you can find color image in the in the internet, right? That's how the ImageNet dataset was captured. But in our case we had to capture real data in real environments and there are limits, of course, the variability in the data. So we had to do some sea trials, so we didn't want to, for example, we didn't want to put the robot like at sea, because if we lose the robot, it's a lot of money that we lose and then the sonar costs like 80,000 Euros. So we tried a surface vehicle, with the sonar under it, like half a meter under, so we could see the the bottom of the lake or the bottom of the river or the ocean, but what happened is that the waves moved the surface vehicle and they moved the sonar and the images were not usable. And the reviewers of my thesis asked me about it, because I mention it like slightly and they wanted to see like images of this, because people usually don't know that. So because the sonar sensor uses sound, so yeah, uses sound beams, 120 sound beams and these sound beams they interfere with each other. So the images look completely wrong. So any kind of, if you need to capture real data, you use kind of weird sensors, there's always problems with like how do you actually use the sensor or properly? Because the manufacturers they usually don't tell you this.

**Interviewer 1:** [00:14:29] Okay. So you had these not good images. Did you have to repeat the process to get the...

**Interviewee:** [00:14:37] Yes. I had to do it again in more like nice conditions in a water tank that we had at the lab. Yes.

**Interviewer 1:** [00:14:44] Okay.

**Interviewee:** [00:14:45] And this of course limits how much data we have we were able to capture, so I have a data set of around 2,000 images, which is not that much, but still better than nothing.

**Interviewer 1:** [00:14:55] Yes, of course. Okay, that's very interesting. So to train your network, did you have to pre-process these images somehow? And do you remember cases when you kind of try to train the model, but you could not, because you haven't done some necessary pre-processing step? Like bugs related to pre-processing.

**Interviewee:** [00:15:16] Yes. Yes. I mean the first thing is that you have to do pre-processing, like to normalize image. Like, for example, if they have pixel values between 0 and 255, you have to divide by 255 and that's also something that people in StackOverflow, they usually forget.

**Interviewer 1:** [00:15:33] Okay.

**Interviewee:** [00:15:34] Or that they, normally there is training set and the test set. But when they test a new image, that they have found from the internet, and you also have to normalize it with the same parameters of the training set. So, for that, you use the wrong parameters and you don't get the correct answer. Yes.

**Interviewer 1:** [00:15:48] Okay.

**Interviewee:** [00:15:51] And I also had issues when by mistake I forgot to put the labels of the training set. So I was getting 100% accuracy because all the labels were the same. So, yeah, kind of a pre-processing mistake because I was transforming the images from sonar files to like hdf5, so I can download it in Keras. And yep, that also happened to me and I was wondering why do I get so much higher accuracy? And it was because all the labels were the same. Yeah. It happens.

**Interviewer 1:** [00:16:19] Okay. Okay. Do you remember any other problems related to training data?

**Interviewee:** [00:16:26] Well, in object detection it is way more difficult to pre-process training data. So we had issues like large datasets.

**Interviewer 1:** [00:16:39] Okay.

**Interviewee:** [00:16:40] We will try and last year to train a model on the OpenImages(?) dataset, which has I think 10 million images or something like that. And we had to have like special hardware to, I mean we have to buy an SSD, because reading hundreds of megabytes of images at the same time in from the drive, it took a lot of time. So it made the whole process slower. So we bought the SSD and it took like 12 hours per epoch. Yeah, that's, I mean you need to have proper hardware if you use large datasets. And now we're using like a supercomputer cluster to do.

**Interviewer 1:** [00:17:11] Okay. Okay. Yes. Thank you. And did you ever use existing datasets for? okay. Well, you just mentioned that actually...

**Interviewee:** [00:17:19] Yeah, OpenImages, I used MNIST and Cifar10, Cifar100, ImageNet, all of these kind of popular

**Interviewer 1:** [00:17:25] ones.

Did you ever had any problems related to this like popular data sets, MNIST and Cifar10, Cifar100?

**Interviewee:** [00:17:33] In general. I mean like the most popular data sets they usually don't have problems, because there is code already that loads them, for example in Kerass. Yeah, they have a loader for Cifar10 and MNIST and Fashion MNIST, so it's very easy to use them. But once you go into the more esoteric datasets, like newer dat sets, then usually you have problems. For example, we try to, we're still trying to use the Coco(?) API, to evaluate into Coco dataset, but we wanted to find an implementation of the MAP metric, this mean average precision, and Coco API has one and it is used by Tensorflow object detection API and so on. But we wanted to use it with our own implementation of the detector, and it works, but now we don't get the same metrics that, I mean, we don't get the same values that other APIs gave us, so we're not sure which one is wrong. So that's also problematic. I mean, we used a Coco API, because we think it is kind of trustworthy, but we are not sure actually. Because sometimes I mean it's made to be used with the dataset and not with something else. So we had to hack it somehow. So we can input our own labels on our own data to be able to use it. So it's not kind of, it's not meant to be used with the other datasets.

**Interviewer 1:** [00:18:49] Okay, thank you. So another question I have is about the model structure. Did you ever face problems related to the wrong, let's say, model structure? Like, I don't know, the types of the layers, the number of the layers, the dimensions of the layers?

**Interviewee:** [00:19:12] Yes, yes. I had this problem as well. You just you need to tune your hyperparameters and not many people know how to do that, like properly. So at the beginning of my PhD in 2016, I think, I was trying to do the program of image matching, that you give two images to the neural network and it tells you if they're the same object or not. And they can be like very different views. I was trying to do this with sonar images. And there's no state of the art for this. I mean, I couldn't just like pre-made model, so I had to make my network structure using kind of research. So, I tried a lot of combination of neural networks, until I found out one that worked, but I also tried like the standard neural networks and they don't work very well. There's like a lot of kind of hidden information into hyperparameters of the networks.

**Interviewer 1:** [00:20:01] Okay, did you...? Do you have a question, Interviewer 2?

**Interviewer 2:** [00:20:08] The small one... But maybe some other problems related to this, while you were trying to achieve to find this perfect model structure that works for your problem. Maybe there were any, some specific, maybe you remember some specific problems, you know when you use some particular layer type, but then you realize it wasn't the best one.

**Interviewee:** [00:20:32] Well, I mean about like kind of layer types, one issue we had is that Tensorflow and Theano, they don't use the same format for inputting the data. Tensorflow uses channels last, that you put the channels at the end of the array and then Theano uses channels at the beginning.

**Interviewer 1:** [00:20:51] First, yes...

**Interviewee:** [00:20:52] Yes, and so for example, I was at the beginning, was using Theano, because Tensorflow didn't exist at the time. So everything worked, bug then if you move the code to Tensorflow, you need to change this, or if not then you get of course wrong. Yeah.

**Interviewer 1:** [00:21:05] Okay.

**Interviewer 2:** [00:21:08] I remember somebody having the similar problem in one of the issues we analyzed, yeah.

**Interviewee:** [00:21:14] Yes! For example, if you have a trained model, using one of the configurations, how do you move it to the other configuration? It is not that easy, because then you need to do some mathematics, and then reshaping of the arrays, and then how do you make sure that it's right? Because you can try whatever combination you want to reshape, but then you need to train to know, I mean not train, to evaluate the number and make sure that still works well, right? So, in general, there are many issues with validation of neural networks. Like you make sure that the neural network is working like what you think it should be working and not in another way.

**Interviewer 1:** [00:21:52] Okay, thank you. I wanted to ask if you had any issues related to hyper... [Break in connection]

**Interviewee:** [00:22:01] Sorry?

**Interviewer 2:** [00:22:02] She was asking whether you had any issues related to hyperparameter tuning, but I guess she has disconnected from us. So one second, I'll just try to add her back. Okay? [Pause] So, let's continue, she will try to rejoin. So, yes, about the hyperparameters tuning? So by this we mean batch size, activation function, selection of activation function, learning rate. Whatever comes to your mind, any problems/bugs with you. Oh, she is here.

**Interviewer 1:** [00:23:26] Yep. I'm back. Sorry.

**Interviewer 2:** [00:23:28] Just in time!

**Interviewee:** [00:23:31] So she was telling me about hyperparameter tuning. Well, I wouldn't call it a bug, but if you use the wrong learning rate, of course, then the network doesn't train. And that's kind of the common bug, but there are also, kind of, I will not call them bugs, but kind of small issues. Like for example, if you're trying [inaudible] network, in the paper they train it with stochastic gradient descent and it works well. But if you want to train the network with, for example, with one of these advanced optimizes like Adam, it doesn't train at all. I mean, that's more like a theory issue, and people don't really know why that happens. I mean you need to tune like the hybrid parameters of the optimizer like this epsilon to make it work and that's something which I mention happen. Another issue we had with hyperparameter tuning like in general, so I had a student that was doing hyperparameter estimation using grid search. And we had trouble, because the like the job in the cluster, after a couple of days it was getting out of memory. And that's because you have to train many combinations of models, and it's like you train one model, and then you discard the model and create a new one. And all of these models were accumulating in the TensorFlow graph and you have to use this function "clear session" in Keras, so you actually start with a new session and the memory doesn't accumulate. That's also something nasty. And it's complicated, because usually it takes many days to tune hyperparameters, and then you get an error like after two days. So, it's like you can fix the errored maybe, but then you need to wait two days to find out if the error is there or not.

**Interviewer 1:** [00:25:17] Okay, thank you. I wanted to ask about the loss functions. Do you usually use a predefined or a custom-written loss function? And did you ever faced any problems because of you selected the wrong loss function, or because it was implemented incorrectly.

**Interviewee:** [00:25:37] Well, in general the loss are implemented correctly. I've used these predefined losses. But if you use a wrong loss, of course, the model trains, but the performance maybe it's not the best or the output makes no sense. Yeah. So yeah, I mean it's a very common mistake that you use the wrong loss function and usually it's hard to notice something, you need to look very closely to say "ahh yes, I wanted to use categorical cross-entropy, but I am using binary cross-entropy", because it still works and there's no error. But the problem is that it is training for a different objective. And as I mentioned before that, we had problems implementing custom losses. Usually because it's not clear how the loss has to be implemented in Tensorflow. And because it's hard to debug in terms of running Keras. I think it is a big issue we had with custom loss functions.

**Interviewer 1:** [00:26:28] Okay, and you mentioned about hardware, that you did not have a good enough hardware to load a lot of images and etc. So I wanted to ask where do you usually train your models? And did you ever have any other problems that were related to hardware?

**Interviewee:** [00:26:49] Well, in general, like when we are developing we trying it in like in our own laptop. What that's only like, we train only for a couple of epochs, but not complete. And then we when we want to train the whole model we usually do it in a workstation, like at office, and if not in a cluster that we have, that we can submit a job to train the model. And usually the problem we have is that the cluster is shared, so sometimes you have to wait for GPUs to be free. It also depends on how big is the model, like in numbers of parameters. Because when we try with large models, we usually need to use like the most expensive GPU, because they have a lot of RAM, and sometimes when they're not available or we don't have these GPUs. Like what was the number, the TeslaV100 ( (??) ) that has 24 or 32 gigabytes of RAM. So we had to get one of these to train like a really big model. Because, the amount of GPU memory constraints the batch size we can use, and the batch size defines like the computational performance, because if you put a large batch size, the network will be training faster. And if you use the batch size of one, then it will take a lot of time to train the model. It's less efficient. I think that's kind of the biggest issue we have in terms of hardware.

**Interviewer 1:** [00:28:09] Okay, thank you. So, do you have any question, Interviewer 2?

**Interviewer 2:** [00:28:15] You just read my mind. So usually when you train your models in the cluster, you must probably use multiple nodes with multiple GPUs at the same time. So any problems related to concurrency, to the using multiple GPUs at one time?

**Interviewee:** [00:28:34] Yes, well, when we generate we don't use multiple nodes, because we don't have like a distributed implementation. Because Tensorflow can do it, but Keras cannot do it like on its own. So we use multiple GPUs and one problem we had is that Keras has this multi-GPU implementation, so you can choose multiple GPUs for computation, but not for memory. Like if you have one GPU with 12[not sure about the number] GBs of RAM and the other GPU with also 12[not sure about the number], you have, you can still limit, because what it does is that it duplicates the model in each GPU, so you don't get more RAM, you get only more computation.

**Interviewer 2:** [00:29:11] I see.

**Interviewee:** [00:29:12] And for this there is a trick that when you implement the gradient computation, you can also fit with like with a batch size of 1. So you can do a forward pass on gradient computation with one batch and then accumulate the gradients and then do the [inaudible] step. And this is not kind of officially supported by Keras, but you can find the code in in many GitHub issues. And this is, it would be nice to have it implemented in Keras, so you can use it with a parameter. So in that case you don't have to use multiple GPUs, but you can train with a much bigger batch size, like virtually. So that is more like an implementation issue, because it is a trick that not many people have, because people just buy more GPUs, more hardware instead of having this problem.

**Interviewer 1:** [00:29:59] Thank you. Okay, so I have one rather classic question, which is what is the most interesting bug that you have faced in your experience so far?Or I don't know the most nasty one that took a lot of your time to fix?

**Interviewee:** [00:30:18] What I can tell you, about one of my students had, because we had it for a very long time. Let me remember, because he was implementing the SSD detector and it seemed to work. But the MAP that he was getting was like a five points lower than the MAP of the paper and the MAP of the repository. And it took a long time to find out why and I think it was a small mistake. There were several bugs: one in the loss, one in the data generator. Because when you do object detection, you don't do, it is not the same as image classification. So you have to pre-process the images to keep bounding boxes and to assign the output bounding boxes of the detector [inaudible]. And I think some of the bounding boxes, like I don't know it's like 8,000 bounding boxes pretty much. And I think 1,000 of them were having the wrong place. So they were kind of being output randomly with some class sometimes, most of the time, they were like background class, which is the correct, but sometimes they were like having random classes. So you could, and, this meant that the detector was been trained to output like random objects at some point. And you could see these objects like popping up in inference. So yeah. So implementing object detection is not easy at all, because there are little details such as one of them is wrong and then the performance is not as good as it should be. I think that's the most nasty, because it took, I think, three month to find out in the end what the bug is.

**Interviewer 1:** [00:31:51] Okay, we feel sorry for your student.

**Interviewee:** [00:31:56] We share the same office and he's trying to implement a post-estimation system and he has this problem with that. When we are using synthetic images by generating them when with Blender and now I told him like why don't you do something that is faster like a renderer that uses OpenGL. And It works, but the thing is he doesn't get the same performance as using Blender and we don't know why. We don't know what the differences is in what blender does and whatever the renderer does. And that's of course, I mean we haven't figured out yet. So that's also another bug.

**Interviewer 1:** [00:32:31] Good luck with this.

**Interviewee:** [00:32:32] Thanks you.

**Interviewer 1:** [00:32:33] Okay. Do you have anything else to add? Like if you remember there any other kind of problems that you had?

**Interviewee:** [00:32:40] So far, no. Well also that the labels were wrong. That was also nice. They need, it took like two weeks to find out, oh, yeah, I forgot to add the labels to the array, so they are all zero. Yes. That also happened to me.

**Interviewer 1:** [00:32:57] Then I guess we can wrap up here...

**Interviewer 2:** [00:32:59] One little question, sorry..

**Interviewer 1:** [00:33:01] Yeah, go ahead.

**Interviewer 2:** [00:33:03] Yeah, you told us about when you convert the model from one framework to other framework, and you do the reshaping, and many different things and you have to validate the conversion in some ways. Can you, please, tell us some more details how you got a date and you convert one model.

**Interviewee:** [00:33:21] Since I got a dataset, so I knew that I trained in one dataset for sonar images and I had to test it. So what I did is that I validated that it had the same accuracy as in the either tested. So I mean you had to record the exact accuracy that they got into a previous run, and then I tested again and I got the same number. So yeah, that's a good validation.

**Interviewer 2:** [00:33:42] Okay, thank you.

**Interviewer 1:** [00:33:45] So thanks a lot. Thank you for your time.

**Interviewer 2:** [00:33:47] Thank you very much. It was very nice. Lots of interesting bugs!

**Interviewer 1:** [00:33:51] Yes. Okay have a nice evening!

**Interviewee:** [00:33:55] You too. Take care. Bye bye.