1:25

Alexey

**This week, we'll talk about developer advocacy engineering for open source projects. We have a special guest today, Merve. Merve works as a developer advocacy engineer at Hugging Face. This is actually not the first time that Merve appears as our guest. Previously, she gave a talk about building a chatbot. I think it was one year ago. The talk is really good, so check it out. Welcome back!**

1:49

Merve

Hello, I'm really happy to talk to you. Every time you have a really nice energy. I really love that. It's usually like a chat rather than just podcasting, to be honest.

# Merve’s background

2:02

Alexey

**[chuckles] Thanks. Okay. But since we're on a podcast, let's start with your background. Can you tell us about your career journey so far?**

2:12

Merve

Yes. I studied industrial engineering, and in industrial engineering, you have mostly operations research-type of stuff. It's like a mix of mathematics, statistics, and coding, to optimize workflows and everything. Over there, I have taken a data science class. I was previously doing forecasting already, but I have taken a data science class and I was like, “I'm going to do this as a job.” Then I started going to boot camps, doing open source projects, I sometimes did Kaggle, I took online courses – I kind of improved myself. Then, when I found my first job as an NLP engineer, I was doing chatbots and question-answering models.

In both of my previous jobs, I was actually doing information retrieval and chatbots mainly. I was using Hugging Face back then and I was already contributing to Hugging Face as an open source contributor. I was already a fan of the company and then people reached out to me saying, “Hey, would you like to work with us?” And I was super happy when that happened. [chuckles] And also did a Master’s and I took part in Google’s and AWS’s community, giving workshops on predictive analytics and NLP and other things – TensorFlow, SageMaker. So far, this is what I did, I would say.

3:58

Alexey

**How did you end up working on NLP stuff? You were doing boot camps, and Kaggle, but then you eventually started working on chatbots. Was it accidental?**

4:12

Merve

In my boot camp – basically, I was going to boot camp sponsored by Microsoft. Over there, I was actually mentoring because it was half theoretical and half practical and I was doing well in both. But I did a project about some text classification and then I stuck to it, and did even more NLP projects. My first ever NLP project was actually at school – I was learning data science with R. It's quite surprising to be honest.

4:51

Alexey

**You did NLP with R?**

4:52

Merve

Yeah. [laughs]

4:53

Alexey

**Okay. [chuckles]**

4:55

Merve

It's quite unexpected, but there is even a TensorFlow for R. You know – if you want to use that.

5:00

Alexey

**Ah, and there is also Keras for R, right?**

5:02

Merve

Yeah. Basically, we have scraped the data sets from Twitter on climate change – people's opinions and everything – and we did sentiment analysis. We sorted a topic modeling in the first place, looked at the embeddings and other stuff. That's how I got into it, I would say. I was like, “This is super cool that you can analyze a lot of people at one place.” And that's how I started doing NLP. One thing after another, I started doing it for a living. [chuckles] Yeah. But also, because I was always working in startups, I was doing everything.

I was taking the data, or getting it from APIs or scraping it. I started from that to EDA, and then building models and even deploying them, which is very end-to-end. Because that's what you do if you're a machine learning engineer working in a startup – you do most of the things. I was also doing predictive analytics, like churn or sales prediction. So yeah, I was basically doing everything. [chuckles] But I did learn a lot of stuff. So I am not regretting that.

# Merve’s first contributions to open source

6:30

Alexey

**You said that while working on NLP, with chatbots, you contributed to open source. You contributed to Hugging Face – I guess also to other libraries. How did it happen for you? What was your first contribution? Do you remember?**

6:45

Merve

Yeah. Basically, how I met Hugging Face was different. They have a chief scientist, Thomas Wolf, and he has a video called “The Future of NLP,” which for two hours or something along those lines, he goes from the start of the NLP and through so many papers, he just analyzes the state of NLP and explains the papers themselves. I was like, “This is so much work. What are they doing?” And then I learned Hugging Face. At my job, I started using Hugging Face as well, especially the birth model and everything.

Then one day, I saw that Thomas Wolf tweeted that they are going to have a contribution sprint about the datasets library. In the datasets library, we actually have something called canonical data sets, which is like – I don't know if you've heard about it – but it's like GLUE or ImageNet. You have to make them easy to use, and to do this, you need to write scripts on these data sets so that it's easily loadable and fed to models in a very native manner, rather than just taking a CSV data set and just dealing with it.

8:10

Alexey

**In SciKit Learn type of [cross-talk] dataset.**

8:13

Merve

Yeah. But datasets are very complicated. For instance, there's something called attention masks in the data set – we have… segmentation data sets are very complicated. So they need to be made easy to use, for instance, named entity recognition, or question-answering data sets – they have span indexes and other stuff. So we were writing scripts to do that, and I contributed a couple of them. That's how I met my colleagues as well.

The other day, I was talking to Contran, who is like the lead of the datasets library, and I was like, “I didn't even know that I would be working with you.” Because I was bugging him back then a lot because I had zero idea about CI/CD or – I mean, they were using CircleCI, for instance. I have never used CI/CD, because I was already working in a very small company. We did not have any development processes that help you maintain big code bases. It wasn't that big.

I learned formatting and everything from Hugging Face. So it was really nice, actually. Then I attended their speech sprints where you fine tune speech models, with the language specific data sets. It was also fun. They were asking me to join and then I joined. It was also that In Google I/O last year, I was talking about transfer learning and I included Hugging Face in my slides. I looked back and I thought that I was sort of destined to work there or something. [chuckles] So, yeah.

10:17

Alexey

**But was it your first open source contribution? I guess not – you probably contributed to other libraries before. Did you?**

10:23

Merve

You know, like issues and other stuff? Not much [cross-talk]

10:28

Alexey

**Not contributions, okay. [cross-talk]**

10:31

Merve

Because libraries like Hugging Face or SciKit Learn have sprints in which the maintainers spend time to help you out in your contribution. Because we observe that once you onboard that contributor for the first time, it's easier for them to contribute later on. It's actually a good thing to have more open source contributors and help people out so that they aren’t scared to contribute.

11:03

Alexey

**Yeah, I imagine that it can be quite scary – quite daunting – when you see all these issues, all this code base and you mentioned things like CI/CD test, code formatting… then you think “Is too much for me? I don't know how to start.” Right?**

11:19

Merve

Yeah, exactly. Now I'm doing PR reviews and stuff. It's actually like a weird journey, you know? You eventually become that person that is giving the review.

11:33

Alexey

**Do you remember – when you were actually making these contributions, you already worked at a startup, right? Were you doing this as part of your job or was this something more like a side activity?**

11:45

Merve

No, it was more like a side activity. I don't think the companies would actually do that unless they are a very big fan of Hugging Face or something.

11:55

Alexey

**But let's say you’re working on something, you use Hugging Face’s library, and then there is a missing feature. Then it makes sense to contribute to the library, right?**

12:05

Merve

Yeah, exactly. For instance, we have our TensorFlow developers, and I see that sometimes they develop to contribute to Keras or TensorFlow in order to ease the process and just optimize some of the functions and workflows that aren't really optimized over there.

# What Merve currently does at Hugging Face (Hub, Spaces)

12:27

Alexey

**Okay. So you started, you contributed to the datasets functionality of Hugging Face before joining, and then they saw you and they offered you a job, right? What do you work on now? Is this datasets part still something you’re also working on?**

12:46

Merve

I have a couple of projects. Basically, the reason why developer advocacy engineering is called “engineering” is because it depends on the company how the job scopes and their technicality changes. And in Hugging Face, it's a very technical job to actually become a developer advocate. We do not have a community builder type of people, but more like a horizontal engineer that is supporting teams and helping people out in general. I wanted to become that person, sort of.

Earlier, I was being interviewed for a machine learning engineering role, but I was just onboarded for my previous job, so I couldn't do that. And then I was approached for this. I was approached twice and I was super happy because I wanted to actually have a position that’s sort of technical, but also developing things for people to have an easier journey in machine learning.

Basically, I have a couple of projects. One that I’m doing is something called Hugging Face Tasks to write in YouTube. As a previous machine learning engineer, I have observed that so many software engineers want to build machine learning products but didn't know where to start. And these Tasks are actually giving the baseline information for a given task like image segmentation or question-answering. It's sort of like I have gained so much know-how in my previous jobs that I wanted to channel it so that people would have a lower entrance level in starting to do machine learning products.

At Hugging Face, we have Hugging Face Hub where there are so many models that you can actually use directly without training your model yourself. It's a bit developed in that manner. This was my first project. I have maintainers and transformers, but on the TensorFlow side, because there are so many people using PyTorch – There’s not many TensorFlow people.

15:34

Alexey

**I thought that Hugging Face uses PyTorch exclusively – that they don't like TensorFlow at all. This is not true, right?**

15:42

Merve

I would say this is not true, but there are a number of people who like PyTorch and Fast.ai – more than people who use TensorFlow and SciKit Learn, I think. They only had one TensorFlow maintainer, Matt. Before we had more TensorFlow maintainers, I was helping Matt out to develop stuff and debug things. We now have more TensorFlow people.

I also integrated Keras into Hugging Face in which, when you host the Keras model on the Hugging Face Hub, you can just push your model with one line of code. It generates a model cart for you, which has insights regarding your model – your model’s architecture, hyperparameters, anything for reproducibility, basically. Hugging Face Hub is sort of all about versioning your models and data sets.

16:45

Alexey

**Like a model registry, right?**

16:47

Merve

Yeah, like a model registry. Most of my job is actually working on Hub. I developed stuff for Keras that would improve the reproducibility of the experiments, version, the models – you can host your Tensor Board inside the model repository, you can have model architecture, metrics, history, etc. in the repository. It's good for collaboration with the teams because if you have your model on your local, it's hard to collaborate with people. It's a bit like GitHub or GitLab but for machine learning, I would say.

17:32

Alexey

**Yeah, that's why it's called Hub as well, right? Hugging Face Hub. Like GitHub – Hugging Face Hub.**

17:37

Merve

Yeah. We also have something called Spaces, which is something where you can just build your demos with Streamlit, Gradio, or just Static, and just share them with people. And recently, we opened a feature called Community Tab, which has pull requests and discussions like you do on GitHub, but for model repositories, or data sets, or Space repositories.

# What is means to be a developer advocacy engineer at Hugging Face

18:07

Alexey

**Yeah, I’m just wondering – you probably cover the engineering parts, right? Everything you described – all these features – they are quite heavy on engineering. You actually need to write code there, make tests, make sure that the CI/CD is working, and all these things. What about the first part – developer advocacy part? Do you also do something like that?**

18:31

Merve

Yeah, we also do that. Basically, the last thing I'm working on, currently, is putting the tabular data modality on the Hub, which is improving reproducibility and collaboration for the tabular data related workflows, having better integration of SciKit Learn stuff, but it's also – basically, everyone in Hugging Face is sort of like a developer advocate. If you look at the Hugging Face course, for instance, every engineer is in the course producing content, shooting videos or doing community sprints – community events.

So everyone is a bit of a developer advocate in that sense. Part of my job is to help people out in the forum, reproduce their errors and fix them. If there's an issue to be opened, I test everything to make sure it's good for developers. I usually try to understand the user journey in everything and I stress test everything, or develop something that would ease the developers’ pain. So it's usually about developer experience, I would say.

I also do something called Keras Sprints, where we serialize the examples on Keras’ official website and we build demos over them, and we later contribute them to Keras. Those examples are very minimalist, for a good reason, because I have talked to François Chollet and he doesn't want it to be overwhelming. There are rules to contribute examples and stuff. So we put models and the demos over there to improve reproducibility over that. Because it's not good to make it too complex – like you go to Keras’ website and you have to run a whole collab in order to see what the model actually does.

So we actually do this for people and host those examples. We did the same for PyTorch, as well. So we have community events like this where we onboard people to contribute to open source as well. I also do workshops on transformers, or building spaces. It's more like a beginner level workshop. I would say that that's also an advocacy part of my job.

21:21

Alexey

**Would you say it's divided 50/50? Like 50 on the advocacy part, and 50 on the engineering part? Or is it something else?**

21:28

Merve

It depends. Currently, we do not have many people working on the tabular data modality – we only have Adrian, who is one of the core contributors of SciKit Learn. We hired one more person who has a famous package on SciKit Learn. Because it is lacking, I am currently coding stuff. But it also depends on developer conferences and everything like that. I think around this season of the year, there are more developer conferences, so I go. Like, next month, I'm going to EPFL, for instance, to present.

Today, I'm going to PyData Paris – I have a couple of things scheduled. So yeah, it depends, but I would say I'm 60-70% coding stuff and like 30-40% presenting things or doing community sprints in order to get more contributors. I'm spending time on the forum or GitHub issues to help people out as well, which is a part of advocacy, I would say.

23:02

Alexey

**Yeah, and I guess the main difference between a dev advocate and a dev advocate engineer is the engineering part, right? So maybe, in the traditional sense of this role, they may spend less time on the actual features of the product or the tool of the product, and they spend more time educating or helping the community. But here, you're doing both, right?**

23:26

Merve

Yeah. Basically, in some companies, it depends heavily on the company. In some of the companies, some of these developer advocates are focused mainly on doing community events, or doing podcasts or educational material. But it's in other companies like, in Hugging Face or in Google as well, we develop stuff inside and we test things. Lately I develop more, but it just depends.

I would say the reason why we call it engineering was that, previously, it was actually called developer advocates, but we received applications from people with lesser technical backgrounds, so in order not to steal their time we have turned the title into engineering because we want to have former MLA engineers that have been doing open source.

The most important thing that we are looking for is already existing open source experience. Because that's the fundamental thing we do. For instance, I do hiring sometimes for the team, and the first thing that I do is look at the GitHub profile of the person.

# The best way to get open source experience (Google Summer of Code, Hacktoberfest, and sprints)

25:04

Alexey

**What's the best way to get this existing open source experience?**

25:09

Merve

You can join the sprints of SciKit Learn or Hugging Face.

25:16

Alexey

**This is how you got this experience, right?**

25:19

Merve

Yeah, exactly. You can pick a library and just go and pick one of the good first issues and assign it to yourself. You open up ER, and it’ll be your first experience. What else? Let me think. Yeah, I would say the good first issues are a good one. Sprints are good. If you want to do code contributions – because we first look at the code contributions to make sure that person is actually technical – but a couple of other things that you can do to actually contribute to open source is not code, but things like documentation, helping people out in Stack Overflow, or forums, writing blog posts or things like that, or submitting bugs or issues, it's also a very valuable thing.

Even developing your own library that solves a problem is a thing. [chuckles] I get these ideas of libraries all the time, although I do not really have any time. I really like building tools in general. I was previously a person building models, but after, I started developing more open source. That’s the thing, I just want to build tools. I became sort of more like a software engineer, I would say, rather than a data scientist now. But yeah, it's fun.

27:01

Alexey

**There are also things like Google Summer of Code. This is similar to Sprints, right? But I guess it takes longer, usually. In the case of Google Summer of Code, I tried to take part. I wasn't accepted, but I know a bit about the process in general.**

27:19

Merve

You weren’t accepted? When did you even apply?

27:23

Alexey

**It was long ago. I think they just didn't have a lot of places for Google Summer of Code. The project I chose was Apache Flink – it was before they became an Apache project – and I think they had just one or two open spaces. [cross-talk] Right now, they have a lot, of course. But it was before they became an Apache project. Yeah, I just got unlucky. But I remember the processes.**

**You need to write some sort of proposal, like what exactly you want to work on. Then if this proposal is selected, you get a mentor and you actually work with this mentor. At the end, you end up contributing a relatively large feature.**

28:09

Merve

I was moving while the applications were open, so I couldn't really apply that time and I regret that. Maybe next year. [chuckles]

28:19

Alexey

**Right now it's open to everyone, not just students, which is even cooler, right? Back then I was a student, so I thought, “Okay, this is my last opportunity.” I was graduating that year, so it was my last opportunity to contribute. But now you don't have to be a student to do that. That's pretty cool. And you get some money for that as well. It's not an insane amount of money – maybe you can get a beer on that money. It’s not much, but still. Especially, if you're a student, that’s…**

28:47

Merve

I do it for the glory at this point. [laughs]

28:52

Alexey

**It's a good bonus, right?**

28:53

Merve

Yeah, exactly.

28:54

Alexey

**Also, there is a thing called Hacktoberfest. I think maybe the first one was last year. Have you heard about this?**

29:03

Merve

Yeah, but I didn't really contribute to Hacktoberfest.

29:09

Alexey

**Yeah, but it was more “global” than Google Summer of Code. I think a big amount of libraries tools took part there. So maybe this October, watch out for it if you want to make an open source contribution.**

29:26

Merve

I am planning to contribute more to SciKit Learn. I met their core developers, who are living here in Paris, and they are doing sprints. But aside from sprints, I'm just planning to pick some good first issues. Because I looked at the code base and it seemed really nice to contribute to. You can also learn a lot from the PRs that you get by means of the clean code, like the sustainability of the code and everything. So it's a big journey. I really like working in open source.

# The peculiarities of hiring as it relates to code contributions

30:02

Alexey

**By the way, coming back to – you said that you take part in the hiring process. And when hiring, you look at the contributions of this person – the code contributions. Do you look at contributions to some projects or contributions to [audio cuts out]?**

30:21

Merve

It can be a person's own project – it doesn't necessarily have to be another code base. If it solves a problem or something, it's a good thing. But the thing is, here, we have standardized the development processes. You develop something or like you contribute to something – you fix a bug – and then you go through this whole PR, like the merger and everything. So we expect a familiarity with developing something for a bigger codebase, I would say.

31:06

Alexey

**Yeah, it's not easy. Sometimes the authors have their own vision and that things are done in a particular way.**

31:17

Merve

Yeah. I can definitely say that in open source, there is no ground truth. You will come across a lot of opinionated people about the code bases, or even so many nitpicking of your PR to a point of… a lot of comments. But you learn a lot, and at some point, you get used to it and you understand the way they develop things. Especially if you start at a place where there’s a really small group of people, you definitely might struggle at first. But I don't think there is like a standard way of developing things that everyone would agree on. So it's quite normal.

32:04

Alexey

**I remember contributing to XGBoost – to the Java library of XGBoost.**

32:07

Merve

Cool!

32:08

Alexey

**Yeah, well… wait till the end. They actually didn't accept my PR. So yeah [chuckles] maybe not so cool at the end. This is quite frustrating, right? So because they have their own way – or expectations of the code – and if this code doesn't follow their expectations... I'm not talking about XGBoost maintainers in particular, but in general about open source libraries. So the way maintainers imagine the feature is written, maybe they might just not accept the request and this can be very frustrating. It was actually my second contribution to XGBoost.**

**My first one was accepted and I was very enthusiastic, like, “Yay! Now I’ll do another one.” And then my other one wasn't accepted, and I was like, “Okay, why am I doing this? No, I don't want to contribute to you anymore.” So how does one deal with this kind of rejection? Because they suck, right? Maintainers have the best motivation, because this is *their* project and in the end, they will have to maintain it, not me. I will commit something and then disappear, and then they will have to deal with this code. But for me, as a contributor, that was a bit demotivating. Maybe you have some suggestions?**

33:23

Merve

Yeah, of course. Two days ago, I had to reject someone's PR because… Basically, we save TensorFlow models in a format called Saved Model, which basically has everything. It has the graph, the variables, etc. It's sort of the agreed way of serializing models in TensorFlow and Keras. And with this, you can use the production tools on TensorFlow Extended Ecosystem as well. Some of the models that are very… in the early days of Keras, there are some serialization techniques like HDF5 and some of the models cannot be saved in this format, because of the old ways of Keras.

For instance, there are models that are CNN encoder and RNN decoder, and then you have a sequential model and you serialize them together. Or you have like a gun model – you do sequential and then inside the list, you put your generator and discriminator. For instance, you cannot save this with the Saved Model. So someone opened a PR to enable HDF5 saving and I had to reject that because this is the agreed… This is a design decision being made to make these models easier for production and it is like witchcraft to actually save those models. That's not really encouraged.

So, the one thing I can say – first, open a discussion in the repository, or organization, to see the design decisions made and why the developers couldn't fix that so far, or any experience or insight they have. This way, you know that there actually is a problem and communicating with the core developers actually helps. I honestly do not have much advice for that. Also writing good unit tests to confirm that it works is a very big part of the work, to be honest.

I test every single thing that I write to make sure that it works and that it is compatible with the rest of the ecosystem. Those tests are there to make sure that any new contribution will not break other functionalities as well. So I would say the unit tests are a very good way of convincing the other person to have your code there. I think these are the two big pieces of advice I have. I don't know if I have any other ones.

36:28

Alexey

**I think the point you made about the discussion is a pretty good one. Maybe this is something that you should do even before contributing code like, “Okay, this is my idea. I want to implement it this way. What do you think about this?” And if you get the green light, then you spend time implementing and then writing tests. Because I guess it can be pretty demotivating if you're rejected after the fact – after you wrote everything. It’s much less demotivating if the idea is rejected before you wrote the code, right?**

37:01

Merve

Yeah, exactly. If that person actually discussed with us beforehand regarding why we still don’t save models this way, they wouldn't have to spend any time on it. [cross-talk] We really like people contributing, so we try to reject people in the least discouraging way. That's a good sign that developers actually care about the time you’ve spent.

37:29

Alexey

**Yeah. I think many tools have their own Slack or Discord communities or discussions in GitHub (a relatively new feature). Or it can even be an issue in GitHub repo. [cross-talk]**

37:43

Merve

Yeah, mostly in issues and discussions. Yeah.

37:47

Alexey

**Talking to open source authors, they usually recommend first going to their Discord and then chat there a little bit before starting to implement a feature.**

37:59

Merve

Yeah, exactly.

# Best resources to learn about NLP besides Hugging Face

38:02

Alexey

**At the beginning, if you remember, I told you that there are a lot of questions for you. I think now it's time to come back to these questions. Sorry to keep you waiting, everyone. The first question is, “Outside of Hugging Face, what's the best resource to learn about NLP? Not just theory, but also application.”**

38:23

Merve

Most of NLP is about solving tasks, which is shaped according to your data – and this can be question-answering. What you want to do is determine that first and then pick the task that is suitable for your use case. It can be question-answering, named entity recognition, or part of speech tagging, or anything. Nowadays, most of these are actually solved with fine-tuning models through transfer learning, which is what transformers are used for. For instance, we have a course – I'm going to write it down in the chat we have…

39:12

Alexey

**Please don’t write it in the chat because I think YouTube blocks all links. So write it to me and then I will send it. It shouldn’t block my links.**

39:23

Merve

Okay. I just sent the link to you.

39:26

Alexey

**And then I will now post to live chat.**

39:29

Merve

It's a good one to get started with NLP. You can also check out the Keras examples – I really like the PyTorch examples and stuff – if you want to learn about the practical side. For the theoretical side, I don't think there is much to learn. At the end of the day, it’s seriously just a different form of data representation and solving your problem according to that, so you just learn how to represent and process your data. It's not even like the tabular side, to be honest.

In NLP, what we do is tokenize the text, which means you have a big paragraph or a sentence and you put them into pieces and just match those pieces to some numbers so that your computer can understand that. It's just like pixel values, how they are labeled between 0 and 255. In NLP, we have pieces of text and they are numbers. After that point, it's more about how you represent your data and that's pretty much it. Most of the problems are solved in a very similar way. I would say you can take the Hugging Face course and look at, for instance, in our GitHub, where we have many code examples.

41:03

Alexey

**I think the question was also about things outside of Hugging Face. Maybe the person who asked already knows about the Hugging Face course?**

41:10

Merve

Yeah, yeah. Basically, how can I say…

41:17

Alexey

**Rasa has a good course, right?**

41:19

Merve

But Rasa is for building chatbots in general. If you want to solve problems, it usually goes from transfer learning. There are a couple of libraries you can use to do that, like spaCy is one of them. I think spaCy also has a course that they can use. But, again, most of the time, I come to a realization that it's mostly about data representation. I've read so many books about this. For instance, I read the NLTK book, which is like the most famous NLP book, I think, to date. It was, again, mostly about the data representation and optimizing your neural network.

Today, we have pre-trained models, like BERT or GPT and we just fine tune them on the downstream tasks, like named entity recognition or sentiment analysis. You usually get better results than just training from scratch, to be honest. So that's why I think someone needs to learn about transfer learning in general. Or maybe, if you're starting deep learning from scratch, there's like so many NLP with TensorFlow stuff. I'm going to send that. In Coursera, there’s also another good course taught by Lawrence. I'm sending you the link.

# Good first projects for NLP

43:01

Alexey

**Yeah, thank you. I think at the beginning, when answering this question, you mentioned that you need to first ask yourself, “What do you need to do?” And then pick a suitable task. Or for “What do you want to do it for?” Do you have some ideas about what exactly could be good projects? Let's say, “I want to learn NLP.” And that's pretty as abstract as it can get. So “I just want to learn NLP.” What could be a good first project? Should it be named entity recognition or something else?**

43:34

Merve

I think a good first project would definitely be sentiment analysis, because the easiest representation of data is going through sentiment analysis – you have sentences and labels. It's seriously not much. In named entity recognition, you have the sentences and inside there are spans of text and it’s the same with their labels. The same goes for question-answering. Let me think. For instance, summarization or paraphrasing – these are also hard tasks, most of the time.

In summarization, there are different types. One is extractive summarization, in which you try to pick the important sentences from a big paragraph of text and representing that is also hard. So I would say sentiment analysis and anything that is like all sentences and a label – that is an easy way to get started with NLP. That's what I did as well. I have a couple of GitHub tutorials. I can send them.

44:46

Alexey

**So it's more like classification, right?**

44:48

Merve

Yeah, it's about classification. I have a poetry classification notebook. That is like a tutorial, sort of. I'm going to send that. They are also on Kaggle. I just sent that. There is not much to analyze about text as well, to be honest. It's not like very big tabular data sets. In text, most of the time your features are universal. It's not very specific and the distributions are also not very specific to the data sets like the tabular ones. I sent you my GitHub project.

45:34

Alexey

**Yeah, I already sent it.**

45:36

Merve

This was like the first tutorial I have written about NLP.

45:47

Alexey

**Yeah, thanks. The next question is, “What is the best way for a newbie to get involved with an open source project?” I think we have mostly answered that. We talked about sprints, We also talked about non-code contributions. We talked about Hacktoberfest. We talked about…**

46:07

Merve

Google Summer of Code, good first issues. Yeah.

46:10

Alexey

**Is there anything we forgot to mention? Or we can just move on to the next one?**

46:15

Merve

Yeah, we can move on.

# The most important topics in NLP right now

46:19

Alexey

**Okay. “What are the most important topics in NLP right now?”**

46:26

Merve

Yes, this is a very good question. Lately Hugging Face just got really away from NLP, I would say. But it does vision, multimodal stuff, reinforcement learning – so I am not super up to date with it. Lately, if you have noticed on the internet as well, most of the trending models are multi-models or generative models, like DALL-E. I read two good papers this year. One was Flamingo. I'm going to write it to you. Flamingo by DeepMind. Yeah, it was a great paper. I'm going to send it. It's also solving multiple tasks with one model. And the T0 model by Hugging Face, which is also a multitask model. It's a very big chatbot that you can speak to. I am sending you another link.

I would say it's focusing mostly on generalization without further fine-tuning your models. We call it zero shots. I just want to speak to this model and let it answer me – and this is like a very big trend. There was also Google’s PaLm model – I'm going to send that as well. It's a very good model. I'm usually not impressed by the models anymore, but I was really impressed by this. It was doing arithmetic, quote completion – it explains jokes and stuff. It's a very big model. They benchmarked the skills of the model against the number of parameters as well. When you go to the website, you will see these three. For instance, in 540 billion parameters, it pretty much does everything.

So I would say the latest trend is to just have a very big model that can do anything – any task – but these are obviously not released open source most of the time [laughs], like we do with Hugging Face. Currently, we are training a model – I don't know if it ended, to be honest – we are training a very big model. I think it's released. It has a lot of parameters that I don't remember because that's what the big science team does.

# NLP ML Engineer vs NLP Data Scientist

49:23

Alexey

**Yeah. That's quite a comprehensive answer. Thanks. The next question is, “What is the difference between what you do as an NLP ML engineer and what an NLP data scientist would do?”**

49:37

Merve

To be honest, I don't think there is something called “NLP data scientist”. It's mostly NLP ML engineers. I have this perception that data scientists are mostly people who do exploratory data analysis, visualization, and analytics. Meanwhile, ML engineers train models, optimize the inference time, or deploy them. So the answer seriously depends on companies.

If you're working in a very big company, then your job becomes much more scoped. But if you're working in a startup, then you pretty much do everything – you're like both of them, I would say. [chuckles] So it really depends. I have never seen an “NLP data scientist” role job application ad, in general.

50:35

Alexey

**Yeah, I think at some point of this conversation, you mentioned that there is not so much exploratory data analysis happening in NLP. It's mostly modeling, right?**

50:44

Merve

Yeah. I'll usually analyze the model – the biases it has with specific inputs, like, genders, races and everything. I would say it's mostly post-processing. After you train the model, you do stress tests on the model to see if it's biased or not. So I would say the analysis is mostly after training the model.

# Project recommendations and other advice to catch the eye of recruiters

51:12

Alexey

**“What type of project would you recommend that new data scientists attempt when trying to catch the eye of employers for entry-level data science positions?”**

51:24

Merve

Honestly – anything works, I would say.

51:31

Alexey

**[chuckles] As long as you *have* projects.**

51:33

Merve

Yeah, exactly. Most people don't even do that, so it's a plus if you do it. I think Kaggle really helps. There is a lot of good stuff on Kaggle. I think companies actually keep [audio cuts out] an open source thing or not. In Kaggle, everything is open. Another thing is, at Hugging Face, we have something called Spaces – I have told you about this. Basically, in Spaces, we host your model demos most made by Streamlit or Gradio and it’s open to everyone. I used to use it sort of like a personal portfolio of my models, because I don't think that technical recruiters actually go to your GitHub profile and run your models and try to make inferences out of them. So it's actually good to have a UI of what your model does.

52:35

Alexey

**Not just recruiters – as a hiring manager, I wouldn't do this. It’s just too much time. First of all, I need to have the environment and this is already tricky. Even if you have the requirements.txt file, right? It doesn't mean it's easy.**

52:50

Merve

Nobody is going to run it. No way. [chuckles]

52:53

Alexey

**Like, I need to do Git Clone, and I need to create a virtual environment, then I need to install everything, and then I need to, I don't know, figure out how to run this thing.**

53:02

Merve

Yeah, exactly. Exactly. So it's actually a good thing.

53:05

Alexey

**Even if there are instructions, it will still take 10 minutes of my time, which I might not have. But as you said, it's already hosted. If there is a UI, that's really great.**

53:15

Merve

Also, if I were applying to a new job, and if they would expect me to build something, I would definitely build the Streamlit or Gradio UI and just send them that instead – where you can just run the Python app.py and it just runs. But having an open hosting of these models – it literally takes one minute to upload those files and Hugging Face builds it for you. Or you can also use other cloud providers and stuff.

53:51

Alexey

**Streamlit has some cloud, right?**

53:52

Merve

Yeah. I think it's just very convenient and the recruiters go, “Hey, this person actually does what I am looking for!” Which is already proof of what you can do.

54:07

Alexey

**I remember – at OLX, as a part of our recruiting process, we have a home assignment. And most people just do what [audio cuts out] “Please train a model and then answer a couple of questions.” Very few people deploy this model – and in the three and a half years that I’ve been working at OLX and taking part in the hiring process, only one person deployed this as a team lead application. And that person was hired because it was so nice.**

**I don't know if there is a correlation or causation. Probably he wasn't hired just because of that. But it was so nice that when he sends the email, like “Here is the ZIP archive and here's Streamlit app that you can play with.” And the first thing I did was just click on this, and it was [audio cuts out].**

55:08

Merve

Firstly, as a machine learning engineer, previously, I hated having to build Flask applications for hours just to show it to the client for five minutes and not even take it to production. I could have just done a good looking Streamlit or Gradio application and just given them a link. Secondly, I am currently – for instance, for my Master's project, I was hosting them openly on Hugging Face Spaces and people were incredibly impressed that I actually did that because TAs (or professors) have a really hard time trying to get your application up and running. You know?

# Merve on Twitch and her podcast

55:49

Alexey

**I do know, yes. [laughs] So you're doing streams on Twitch, aren't you?**

55:56

Merve

Yeah, but I gave up because… Basically, I recently moved to Paris and prior to that, in Turkey, my internet was extremely bad. I even had a talk with Abhishek (ISP) and I had to cancel it. I said, “This is the last thing I'm going to do.” Because it is actually quite disappointing. I think I'm going to get back because now that I'm in Paris, my internet is actually stable. I will probably get back to doing podcasts with awesome people. I met really good people here from – I don't know, like SciKit Learn, Big Science, Hugging Face.

56:42

Alexey

**Did you bring your microphone with you?**

56:45

Merve

I'm going to go back to my hometown to bring it. Because it's too heavy. It costs a lot. Initially, I thought that I would just bring my essential stuff and then come back and take it. But I can just do it anywhere, so it should be fine. [laughs]

57:07

Alexey

**[laughs Okay. My next question was about the podcast – your podcast – influence podcast. But I guess this is something you're not doing at the moment [audio cuts out].**

57:20

Merve

Yeah, for a while I gave up because of my internet (again). I had to cancel episodes because of how the Internet was incompetent. But now that I'm here, I'm actually planning to have them physically. For instance, I might just visit Berlin, and we can have a physical podcast session together. [chuckles]

# Finding Merve online

57:42

Alexey

**That would be nice. Tell me when you come to Berlin. Okay. I think we should be wrapping up. Is there anything you want to say before we finish?**

57:56

Merve

For the last month, I was very busy. But, if you have any questions, you can reach out to me through the DataTalks.club Slack directly or my Twitter account. I usually respond.

# Merve and Mario Kart

58:14

Alexey

**Okay. There actually is one question. “Why are you so bad at Mario Kart?**

58:22

Merve

It depends on the speed. Like, at 50 and 100 CC, I'm actually not that bad. But after 150 CC, I am really bad, because it's like a psychedelic trip to actually play that fast. Basically, in the Hugging Face office, people love Mario Kart and we are planning a tournament really soon. They are really good at this, like you cannot believe. I am trying to just improve myself in the meanwhile. [chuckles]

58:57

Alexey

**So I guess that was a question from one of your colleagues?**

58:59

Merve

I don't know. I posted about how I am bad at Mario Kart. So it can be that or from my colleagues. [laughs] Yeah, they bash me a lot.

59:11

Alexey

**[chuckles] Okay. I think that's all we have time for today. I actually didn't ask like half of the questions I prepared, but maybe next time. Who knows? So yeah, thanks a lot. Thanks for joining us today. Thanks for finding time to answer our questions. It is always a pleasure talking to you.**

59:29

Merve

Yeah, you too. See you.

59:30

Alexey

**Goodbye, everyone.**