**Name:** Interview-12-20220805

**Created On:** 8/9/2022 4:49:34 PM

**Created By:** WJ

**Modified On:** 8/13/2022 8:47:52 PM

**Modified By:** WJ

**Size:** 10 KB

Speaker 1:

Okay. So first, I'd like to ask you questions about your background. So the background questions can be answered briefly so we can save time for the other questions. So first, could you briefly describe your role in your team?

Speaker 2:

So I'm currently a PhD student in Department of Software System and Cybersecurity. So basically, I work by myself or either in a group of four to five people, but mostly I code by myself, so.1

Speaker 1:

Okay. So what kind of project are you working on?

Speaker 2:

Basically, I'm working on using a pre-trained transformer model for software security issues, like vulnerability detection or automated vulnerability repair, something like that. Yeah.2

Speaker 1:

Okay. So can you tell me a recent time when you used a pre-trained model from an external ModelHub?

Speaker 2:

Yeah. So previously, I was doing a software security repair project. Then I download a pre-trained model from the HuggingFace ModelHub, and then further fine-tuned for the repair dataset. Also, I upload my fine-tuned model back to the ModelHub to facilitate the future research. Yeah.3

Speaker 1:

Okay. So have you ever retrained any models?

Speaker 2:

What do you mean by retrain?

Speaker 1:

Or, I mean like retrained model from scratch.

Speaker 2:

Yeah, I did try retrain from scratch, like pre-training from scratch. Yeah, in that case, I just need to load the configuration file from HuggingFace and then I started my own pre-training process. Yeah.4

Speaker 1:

Okay. So have you used any model as a backbone?

Speaker 2:

Yeah. I've tried different models, like T5 and the BERT architecture, also the DeBERTa. So most of them are proposed by large company like Salesforce and Microsoft or even Google. Yeah.5

Speaker 1:

Okay. So the next set of questions is related to how you select a pre-trained model. So here we are trying to understand the process that software engineers follow as they decide which pre-trained model to use in their projects. Could you think about the last time when you choose a pre-trained model from ModelHub? So how did you choose it? Can you summarize your decision making process?

Speaker 2:

Yeah. So because my task are mostly related to the programming language, so the search space is actually really small, because there is just a few large pre-trained model for programming language. So when I decide which one to choose, I would maybe read the paper first and also to check their architecture. Then the other thing I would consider is the model size, because there are different kind of size of model, like from millions of parameters to billions. But given that my machine only got two GPU with 48 gigabytes memory, I would prefer the model having parameter between 100 million to 1 billion. Yeah. So the model having more than 1 billion parameters is not suitable for my case. Yeah. So I guess the model size is the main concern known.6

Speaker 1:

Okay.

Speaker 2:

Yeah.

Speaker 1:

So where do you usually select the pre-trained models, whether you select the models from the ModelHub or the models from the GitHub project?

Speaker 2:

Yeah. Mostly from ModelHub if it's available. Yeah. Because it's more convenient and it make my life easier.7

Speaker 1:

Okay. So when selecting the model, do you care more about the model's performance than the architecture?

Speaker 2:

Let me think. Yeah. Because when I retrieved the model, actually I wouldn't expect it perform too well because my domain is more specific. There is no pre-trained model, especially for software security. So mostly I didn't test the pre-trained model first. I'll test after fine-tuning. Yeah.8

Speaker 1:

Okay. So do you think the pre-trained models available in the model registries or ModelHubs, they accurately describe their behavior?

Speaker 2:

Yeah. Most of them. What do you mean by behavior?

Speaker 1:

So there could be some discrepancies between the actual performance and the claimed performance in the documentation.

Speaker 2:

Oh, I see. I see. Actually, I'm not sure about this, because, yeah, I didn't validate the performance too much. So I'm not really sure about whether the performance achieve what they propose in their work9 course. Yeah.

Speaker 1:

Okay. So let's assume these discrepancy exist. To what extent do you think the discrepancies can affect your decision making process?

Speaker 2:

Oh, you mean there if there is a performance gap.

Speaker 1:

Yeah.

Speaker 2:

Yeah. Yeah. I think it would be not a small issue because mostly it could be one of my criteria to select the model. Yeah. If the author, they propose a good performance of their model, I expect the model to be more accurate than the other. I mean, if the pre-trained model is actually more accurate, it would also be helpful during the fine-tuning. So I think it's a big concern. Yeah.10

Speaker 1:

Okay. Next question is to what extent do you think the robustness of the models can affect your decision?

Speaker 2:

Okay. So you mean the model after fine-tuning or the overall?

Speaker 1:

I mean the pre-trained model available in the ModelHubs.

Speaker 2:

Yeah. Oh, you mean the robustness of the pre-trained model in the ModelHub.

Speaker 1:

Yeah.

Speaker 2:

Okay. So what do you mean by robustness in this case?

Speaker 1:

So whether a model can be used in different scenarios, like there could be some adversarial attacks for the model.

Speaker 2:

Yeah. Okay. I see. I think for most of the tasks, the model should be fine-tuned before being used by the user. Yeah. Because from my experience, even a fine-tuned model would have robustness issue. For example, one of my vulnerability detection model, it's really interesting that it takes a function written in C language as input and then the output is the prediction to predict whether it's vulnerable code or not a vulnerable function.11

Speaker 1:

Yeah.

Speaker 2:

But when we use the fine-tuned model on that dataset, and then if we randomly add some space or add some blank lines, it would affect the model prediction somehow for some cases. So yeah. Even the fine-tuned model would have that robustness issue, I think, at the moment. Yeah.

Speaker 1:

Okay. So the next question is to what extent do you think the explainability of the model can affect your decision?

Speaker 2:

Actually, because nearly all of the pre-trained model are deep learning based model, yeah, so in terms of deep learning model... Actually, explainability wouldn't be my main concern, because I think, in my opinion, the explainability aspect of deep learning is still not mature. So even if we add any explainable component on top of deep learning model, I think it's not really useful at the moment. Yeah. But it could be developed better in the future. Yeah.12

Speaker 1:

Okay. So next question is how frequently do you use retrain models?

Speaker 2:

Mostly every week, because I try different models a lot. Yeah. So mostly maybe two to three models per week. Yeah.13

Speaker 1:

Okay. So do you think the lack of trainability or fine-tunability is a problem while using a pre-trained model?

Speaker 2:

Okay. So what do you mean by lack of the trainability?

Speaker 1:

So it's like for some ModelHub or model registries, it's hard to fine-tune the models. So I mean-

Speaker 2:

Oh you mean the model without registry or?

Speaker 1:

I mean, the pre-trained model in the registry is whether they are easy to fine-tune or retrain. Do you think this is a problem?

Speaker 2:

Oh, do you mean in the HuggingFace hub?

Speaker 1:

Or I mean, in any hub you have used.

Speaker 2:

Okay. Yeah. Let me think. So I think in technical aspect, it's really easy to train a model from the hub because the workflow has been implemented really well and then you can run it without any bugs. But if a model doesn't contain any information about what it is, it would be a little bit more difficult to try to understand what this pre-trained model is and how can I use it. Yeah. So I think it would be better if every model can have detailed description, but given that, for example, in HuggingFace, the model page, there is a model card feature, I guess, but some of the author wouldn't fill out model card, just leave it a blank. Yeah. That would be a issue. Yeah.14

Speaker 1:

Okay. So except for this aspect, is there any other challenges you faced before when selecting a pre-trained model?

Speaker 2:

Let me think. No, I think it's all right. Yeah.15

Speaker 1:

Okay. Okay.

Speaker 2:

Yeah.

Speaker 1:

All right. So we'll move on to the next set of questions, which is really to the deeper new software attributes. We want to learn about what sort of information is useful to the engineers who use pre-trained models. Here, I'd like to show you the definitions for three traditional attributes, and this three attributes are quality, popularity and maintenance. These are from the npm, which is used for JavaScript packages. Could you take a brief look at the definition of this three attributes and let me know when you are ready.

Speaker 2:

Yes. All done.

Speaker 1:

Okay. So what do you think would best help you or your teams select a pre-trained model from the model registries for these three attributes?

Speaker 2:

I think maybe the popularity would be, yeah.

Speaker 1:

Okay.

Speaker 2:

Yeah.

Speaker 1:

So is there a reason?

Speaker 2:

So because I think it's the most explicit metrics in each ModelHub at the moment. Yeah. Mostly I would look into number of downloads. Yeah. Yeah. Because I think if it's a popular model downloaded by many users, it could be a good start point to try. Yeah.

Speaker 1:

Okay. So do you care about the quality or maintenance if the ModelHub can provide this information?

Speaker 2:

Yeah. I think it would be great. Yeah, I think it would be useful, but maybe not that useful like traditional software package. But yeah, we still need it.

Speaker 1:

So is there a reason why you say it's not as useful as a traditional package is?

Speaker 2:

Because, I mean, it would be really hard to modify a big pre-trained model because it consume a lot of time and resource. So mostly yeah, like recent GPT-3 three and some large pre-trained model, there are actually some issues during pre-training, but the model didn't actually get retrained because it's just too large. So yeah, it's really hard to modify the pre-trained model. So the maintenance is just not that... I'm not really sure.16

Speaker 1:

Okay. Okay. So here we define three different specific attributes, provenance, reproducibility and portability. I will ask two questions for each attributes here. So first, let's discuss about the provenance here, we define it as a measure of model lineage or traceability. Examples are whether the documentation contains a link to the paper, whether it's from a research or commercial group and this kind of thing. So can you think about the time when you met any provenance problem before?

Speaker 2:

Yeah. Yeah. I have some experience with it. For example, if it's research like six months or long time ago, the external website mostly can't be accessed. Yeah. Because I think, yeah, that the host is just maybe deleted, I guess. For the dataset, it's another issue that like most of the... When we try to reproduce the results of the models, most of the time we cannot directly get the available dataset even though HuggingFace provide the dataset hub as well. But I guess it's not that popular in academic area. I'm not really sure.

Yeah. So this is another issue because when you reconstruct the dataset, there exist some other issues, then yeah, the results could be different. Yeah. For the architecture and for the training strategy, most of the user would report in their paper, but it's actually really annoying that every author put their hyper parameter setting in the model architecture in different ways. So it is really annoying to search the parameters or the architecture in each paper. Yeah. So that's another one.17

Speaker 1:

Okay. So do you mean if the model registries can provide this kind of information, it can be really useful?

Speaker 2:

Yeah. It can be. Yeah. It can save a lot of time because currently some author prefer to report hyper parameter in text, but some of them report in the table. Then I think most of them didn't completely report everything. Yeah. Every parameters and every, yeah, parameters used for the model architecture as well. Yeah.

Speaker 1:

Okay. So-

Speaker 2:

I think it's because of maybe it would be too lengthy to include all of them in the paper. But if they can include in the ModelHub, it would be great. Yeah, definitely.

Speaker 1:

Okay. Okay. So is there any other thing you think would be helpful to know beforehand in order to solve the provenance problems?

Speaker 2:

Yeah. I think the most important thing is, like we discussed, yeah, the training strategy and the architecture parameters, something. Yeah, just to make sure to cover all of the details of the model.18

Speaker 1:

Okay. All right. Then we'll move on to the reproducibility part. So here we define it as the ability of a different practitioner to produce the same accuracy and training or evaluation time from a pre-trained model as defined in the paper, source code or the original group. Can you think about a time when you met any reproducibility problems before?

Speaker 2:

Yeah. Yeah. So actually, the reproducibility, I think it's always a really big problem in research. Yeah, because even that I think some of the research, they would maintain their model and as well as their repository. But there's still some research that even you open issues on their repository, they wouldn't respond to you and, or yeah. What else? Let me think. Reproducibility. I think you mentioned notebook and inference demo here. I think it's really helpful too as a good quick start section.

So I've experienced some of the paper, they're doing this. It's about object detection transformer. Yeah. So in that case, they create a notebook demo of their model with the architecture and because sometimes it would be convenient to use notebook as a demo, since you can learn the architecture from the markdown and then after the markdown it's the code. So it would be convenient. Yeah.19

Speaker 1:

Okay. So is there any other thing you think would be useful to know beforehand in order to solve the reproducibility problems?

Speaker 2:

Variance. Yeah. Actually, accuracy variance could be useful, but given that some of the model, like I said, it's very large. So it would be difficult to get the accuracy variance, given that if you try to repeat experiments multiple time when your model and your dataset gets larger, it would be really difficult. Yeah. But it's actually a good thing to know.20

Speaker 1:

Okay. So the last attribute here is portability. We define it as the ease with which an engineer can take a pre-trained model and reuse it in another environment or another software project. Can you think about a time when you made any portability problems before?

Speaker 2:

Yeah. I think for the fine-tunability, I had some issues for this. But I'm still not really sure about the reason. Yeah, the thing was sometimes I would try using pre-trained language model trained from natural language and fine-tuned for a programming language. For some cases it would work. Let me think. There is a model called DeBERTa, and currently there is only DeBERTa version trained from natural language. There was a time I tried to pre-train the DeBERTa architecture from scratch using programming language.

But the model just didn't work. The accuracy after fine-tuning the pre-trained model from scratch, it's really bad. Yeah. Also, during pre-training that model, I had some other issues, which is I think about the gradient update. So when I try using programming language to pre-train the DeBERTa architecture, during the first few updates, there is a gradient update issue. The gradient, it becomes a nan-value and then the training stopped. Yeah.

But yeah, I tried solving that issue by taking the pre-trained model from the checkpoint provided by the original author, which means I didn't train it from scratch, instead I take the model pre-trained on natural language and then I further pre-trained using programming language. In that case, the gradient update wouldn't be an issue. But the final model is just very bad. Yeah.21

Speaker 1:

Okay. So what do you think would be helpful to know beforehand in order to solve these problems you mentioned?

Speaker 2:

I think this problem could be out of scope of the original research. Yeah, because they are focused on the natural language and the pre-training strategy. But on the other hand, I'm focusing on the programming language. So I think this could be another problem to solve. So I mean, this is not their problem.

Speaker 1:

Yeah.

Speaker 2:

Yeah.

Speaker 1:

Okay.

Speaker 2:

Yeah, but it's still a bit interesting because when we do the same thing for the BERT architecture or other architecture like T5, the issue is not the DeBERTa model. So I think there is something to be investigating that model, but I'm just not sure.22

Speaker 1:

Okay. Okay. All right. The last question for this part is except for this three attributes, do you think there are any other attributes would be helpful for the pre-trained models?

Speaker 2:

You mean the portability?

Speaker 1:

I mean, except for these three attribute here. Yeah.

Speaker 2:

Oh I see. Let me think. So I think if the author can share the information about their hardware and their exact pre-training time, it would be, yeah, a good information. Yeah.

Speaker 1:

So do you think other training details like the last trend would be helpful as well?

Speaker 2:

Yeah. I mean, if can have some information about what kind of GPU and CPU that the original authors used and what's the GPU memory and the CPU memory, it would be a useful information as a reference. Yeah.23

Speaker 1:

Okay. Okay. So we have a last set of questions, which is about the trustworthiness. I think we're running out of time and you can let me know if you have to leave.

Speaker 2:

Oh, no. No problem.

Speaker 1:

Okay. Okay. Sure. So the last set of questions is about the pre-trained model trustworthiness. Here, we are trying to understand how pre-trained model shortcomings affect engineers' ability to rely on and reuse them. So the first question here is which aspects of the pre-trained model do you assume are trustworthy in the model registries?

Speaker 2:

You mean the information in the registry?

Speaker 1:

Yes.

Speaker 2:

So I think the architecture should be the most trustworthy because if the... So if the architecture in the configuration file is wrong, then you couldn't even further run the model because there is a size difference, like dimension. It's not compatible. Yeah.

Speaker 1:

Okay. So do you think the performance are trustworthy?

Speaker 2:

Performance? It really depends. But if it's the same, or really if it's the same task like the original author did, yeah, it's trustworthy. But if it's similar task or other task, I don't think the pre-trained model is trustable. Yeah.24

Speaker 1:

Okay. So have you found any discrepancies between the claimed pre-trained models and the downloaded version in terms of accuracy, latency and architecture?

Speaker 2:

No.25

Speaker 1:

Okay.

Speaker 2:

So yeah. Yeah. Because mostly I evaluate the model after fine-tuning, so I haven't experienced this kind of thing too much.

Speaker 1:

Okay. Okay. So here let's assume the discrepancies exist again. To what extent do you think these discrepancies are acceptable?

Speaker 2:

You mean how much gap or?

Speaker 1:

For example, if the discrepancy is less than a certain percentage.

Speaker 2:

Oh yeah, yeah.

Speaker 1:

Yeah.

Speaker 2:

Let me think. I would expect like 20% would be. Yeah. But it's like a relative 20%, not absolute 20%. Yeah.26

Speaker 1:

Okay. So the last question here is do you think the discrepancies will have significant impact?

Speaker 2:

Yeah. But it depends on the task you want to perform. Yeah. But I think it's a big concern if the discrepancies, if it's large, yeah, it's really big concern.

Speaker 1:

Okay. Okay. So you said it depends which tasks. Do you have any specific examples?

Speaker 2:

Yeah. So I think there are two aspects. So one is for the research academic area. It would be big concern no matter which task it is. Because future work is going to build on top of your work. So it's a really big concern. For the industry, it really depends on the task, how you want to use the pre-trained model. Yeah. But I think at the moment, it's still not really popular in the industry for important tasks.

But yeah, for some other task, like software security, they are not very much AI model cases because it's a really dangerous task that could arise some other concerns if you're adopting an AI models. So I just feel like, yeah, the industry is not ready some of the tasks. But for some other task, like co-completion and co-suggestion like co-pilot, it's not that dangerous like software security, so yeah.

Speaker 1:

Okay. So one more question here. So if you have enough computational resources, do you prefer to reuse existing models or you would like to build your own model?

Speaker 2:

Actually, it depends. But if I got resources, I would definitely try to do things from scratch for one time. But I think most of the time I would still prefer to use a pre-trained model. Yeah. Yeah, because in that case, yeah, it's just saving your time. If you're going to train something from scratch, you also need to consider some other aspects like the dataset and other things. Yeah.27

Speaker 1:

Yeah. Okay. Okay. That's all my questions for the interview. I'll stop recording now.

**Annotations**

1 Role:

- PhD student. Work by myself or either in a group of four to five people.

- Using pre-trained transformer model for software security issues.

2

3 Reuse scenario:

- download -> fine-tune for the repair dataset.

- I upload my fine-tuned model back to the model hub to facilitate future research.

4 Retrain:

- load the configuration file from HF and start my own pre-training process.

Fine-tune:

- Yes

5 Backbone:

- Yes. Most of the models are proposed by large company.

6 Decision making:

- Programming language -> search space is really small.

- because there is just a few large PTM for programming languages

- Read the paper first and check their architecture.

- Also the model size, which is a main concern.

- can be used in my two GPU environment

-

7 Where:

- Model hub

- more convenient and easier

8 Performance or architecture:

- performance

- I wouldn't expect it perform too well because of the domain

- Mostly I test after fine-tuning.

9 Accurate behavior:

- Not sure. I didn't validate the performance

10 Discrepancy:

- Not a small issue. Mostly it could be one of my criteria to select the model

- It's a big concern.

11 Robustness:

- Even a fine-tuned model would have robustness issue.

12 Explainability:

- Nearly all PTMs are DL models.

- Wouldn't be my concern because the explainability aspect of DL is still unmature.

- It's not really useful at the moment. But it could be developed better in the future.

13 Retrain:

- Every week. Two or three models per week.

14 Trainability/fine-tunabiltiy:

- If a model doesn't contain any info about what it is, it would be more difficult.

- It would be better if every model can have detailed description.

- In HF, some of the authors wouldn't fill out the mode lcard. Just blank. That would be an issue.

15 Other challenges:

- No.

16 Traditional attributes:

- Popularity >

- Popularity: most explicit metrics in each model hub at the moment.

- lots of downloads -> could be a good start point to try.

-Quality & maintenance: would be useful, but not that useful like traditional SW package.

- It would be hard to modify a model

- There are some issues during pre-training, but the model didn't actually get retrained beacsue it's too large.

17 Provenance issue:

- The host of external website was maybe deleted

- Most of the time we cannot directly get the available dataset.

- When you reconstruct the dataset, there exist some other issues.

- Every author put their hyper-parameter setting in the model architecture in different ways. It's really annoying to search the params or the architecture in each paper.

18 Provenance help:

- Training strategy and architecture parameters.

- Every parameters and those used for the model architecture.

- It might be too lengthy to include all of them in the paper, but if they can include in the model hub, it would be great definitely.

19 Reproducibility issue:

-There's some research that even you open issues on their repo, they wouldn't respond to you.

20 Reproducibility help:

- Inference demo.

- Notebook demo with the architecture

- Accuracy variance.

21 Portability issue:

- Fine-tunability

- gradient update issue -> nan value -> training stopped

-

22 Portability help:

- for this problem, it's out of the scope of the original research

23 Other attributes:

- Hardware

- Excat pre-training time

- What kind of GPU and CPU

- What's the memory of GPU and CPU

- would be useful as a reference

24 Trustworthiness:

- Architecture should be the most trustworthy

- Because if the architecture is wrong, you couldn't even run the model because the dimension is different

- Performance: really depends

- If it's from the original author, it's trustworthy.

- If it's similar task or other task, it's not trustable.

25 Discrepancy?

- No

- Mostly I evaluate the model after fine-tuning.

26 Acceptable discrepancy:

- relative 20%

27 Significant impact:

- For research, it would be big concern no matter which task it is. Because future work is going to build on top of existing work.

- For industry, it really depends on the task.

- The industry is not ready for some of the tasks.