Assignment 2: Operationalizing Machine Learning

Interview with Erik Svensson

Walter Pach / Vera Svensson
CSCE-585 / Fall / 2022

https://github.com/csce585-mlsystems/MilCiv/blob/main/assignments/Assignment2.pdf

1. Introduction

The development of machine learning models consists of many stages, all of which can look widely different depending on the task, the experience of the developers and the size and dynamic of the team. To understand the various stages, and how developers think during the development process could be an aid in improving as a ML engineer. In a mission to expand our knowledge, we have conducted a small interview study where we have interviewed one ML engineer about the process of developing new ML models. The interview consisted of the control questions, many of which were at times inappropriate to address the interviewee's unique experience. We appreciate understanding in this regard.

2. Methods

In this section the method used to conduct the interview and extract information will be presented.

2.1. Participants

In this interview, we spoke with Erik Svensson about his role and strategy as a ML engineer in a research and development environment. Mr. Svensson is a Swedish ML engineer who works on a contract basis through his own ML consultancy company. Additionally, Mr. Svensson works to educate about ML/AI in an undisclosed research and development position. Unfortunately, Mr. Svensson is unable to disclose both the client and the project that he is currently working on.

2.2. Interview

Since Mr. Svensson lives in Sweden, the interview had to be done over a video call. It was done using Zoom, since it is a widely available software with good support for recording audio. Vera acted as the main interviewer while Walter took notes and asked additional questions. The questions were heavily inspired by the questions used in the study *Operationalizing Machine Learning: An Interview Study*.

2.3. Analysis

The audio recording of the interview was transcribed and then analyzed. The knowledge was summarized in the table below. There were strong limitations from the interview about the tools used and the subject matter of Mr. Svensson's work, but we attempted to summarize it below.

	Data Collection	Experimentation	Evaluation and Deployment	Monitoring and Response
Run	Identify and tag relevant video pieces	Prototyping	N/A	Track the model's performance
	Regular text files, CSVs, etc.	PyTorch, TensorFlow, ResNet, COCO	N/A	Grafana
Pipeline	Regularly scheduled, data formatting, automated process	N/A	Deployments should be automated to production	Scheduled evaluation of defined metrics
	Python Scripts, Video codec decoder	N/A	CI/CD tools, Pipelines	Grafana, custom Python scripts
Component	Review of data, cleaning data	Feature development, model training	Optimize for the production environment, rewrite model	Data Distribution and Visualization
	Homemade Python scripts to review data, manual review	TensorFlow, PyTorch, lots of Python, ResNet	C++, Model cleanup, Model hardening	Grafana, Hard drive distribution
Infrastructure	Velocity	Velocity	Validate Early	Versioning, etc.
	Git, VCS, hard drives	Git, VCS, hard drive versioning	Testing model against novel data	Grafana, logging, etc.

Table 1: Primary goals and tools for each layer of the ML stack

3. MLOPS Used

Mr. Svensson's work current centers around the development of models and training systems for models. This involves a suite of homemade scripts to format, decode, and prepare data. This is then used to train small models and promote a kind of active learning. These speeds up the process and helps to identify patterns in the data faster. Mr. Svensson uses industry-standard version control systems and employs various branch-management techniques but does not use the degree of pipelining that is advocated or encouraged in the course.

4. MLOPS Challenges and Opportunities

The tools used by Mr. Svensson offer a degree of rapid prototyping like that which is done in the classroom setting. However, these are not applicable in the production environment since a lack of defined procedure can introduce uncertainty, variability, and inconsistency in models.

Additionally, the ML operations deals with the problem of sharing large volumes of training/test data, ensuring that it is preserved in its various forms, and the formatting of the data.

MLOPS poses a challenge of standardizing the approach of extremely technical areas (e.g. the training of the model, standardizing of the data, and the optimization of the parameters). However, there are also elements of MLOPS that cannot be standardized regarding speed and rely on human and developer intervention, introducing some of the efficiency of Mr. Svensson's approach (I.e. rapid prototyping).

From our interview, it seemed that Mr. Svensson is deterred from implementing many of the standardization capabilities that MLOPS offers due to the high initial cost of configuration. However, as Mr. Svensson noted, in the rapid prototyping environment, this efficiency of deploying and testing is sacrificed for the flexibility of development.

5. Conclusions

While Mr. Svensson was unable to talk about many aspects of the work he currently does—and in light of the limitation that he currently works as an R&D engineer—we were able to identify various stages of his development and operations lifecycle.

5.1 Development vs Production

Mr. Svensson repeatedly pointed out the significant differences between working with an R&D approach, which he is currently doing, and a production approach. In an R&D setting, the focus seems to be on quickly producing a model, allowing you to test and analyze ideas. He mentions that Python is widely used as the main language as it is easy to quickly set up a model and learn something from it. However, going from that to actual production is a huge step that requires a lot of resources from the company. One of reasons being that, in Mr. Svensson experience, you leave Python and must translate the model to C, C++ or any other similar language. Another difference mentioned is the increased load of testing in production. Of course, testing is a part of an R&D process. However, when moving into production it is suddenly critical, and testing must be done thoroughly.

Furthermore, when moving into production the model must be adapted to the target device, which can pose a challenge. Mr. Svensson points out that when developing an experimental model, you have access to a variety of things that might not be available in the target device, like servers and GPUs.

It is also mentioned that when moving to production, you must rethink your methods of training. Mr. Svensson mentions federated learning, which is a method he thinks large companies like Google and Apple are currently using on their devices. He briefly explained that the models used in large scale corporations are dependent on an enormous infrastructure and require that in order to deploy and re-train using federated learning. This includes an array of continuous integration and automated tests.

5.2 Team Size

Mr. Svensson is currently working in a small team and is highlighting the differences between the tools and structures required to coordinate a large team, compared to what he is currently experiencing. In a large team you have procedures in place to efficiently work on the code, like nightly builds, unit testing and testing on different hardware. He mentions Grafana, which is a tool he enjoys and has been using when working in a bigger team. It is a dashboard software used to provide reports of builds, tests, and commits, which is a helpful way to summarize the information provided from the structures set up in a larger team. In a larger team there is also the process of running automated tests and having a thorough code review every time you

commit new code. Having this, and setting up previously mentioned structures, requires a lot of resources and in a small team, it might simply be a waste. Mr. Svensson currently does not use any other tools than GitHub, which he uses to version control. He also does not spend time setting up any automated tests when working on small projects and is currently doing all his testing manually.

5.3 Moving Data

Mr. Svensson discussed that in contrast to a team that deploys models and infrastructure in a production environment, his company works with their other branches to physically transport models and data in the form of hard drives. He discussed how this approach is very efficient and effective at the scale of terabytes but that it would not scale well as the project grows outside of its experimentation and development

5.4 Bugs

Upon the question of how Mr. Svensson responds to bugs, he admits that he currently does not track them and instead relies on resolving the bugs as they appear. Whenever there is a bug, he simply tries to fix it straight away. He does however mention that when working professionally and closer with a team, tracking bugs is critical and there should be a tracking system in place in order to improve efficiency and communication between stakeholders and the development team.

5.5 Public Models

The use of public models is common. Resnet is mentioned as a popular model. Mr. Svensson is impressed by the performance of the publicly available models. However, he prefers to downscale them to reduce the degrees of freedom. It takes a lot of time but it prevents the model from overfitting as much. When working in an education setting he mentions the use of YOLOv5. Additionally, he mentions the importance of deploying models in an efficient language and format, explaining that infrastructure is commonly in Go or Rust.

APPENDIX A: Questions Asked

1. **Nature of the ML Task**
    - What is the ML task you are trying to solve?
    - Is it a classification or regression task?
2. **Modeling and experimentation ideas**
    - How do you come up with experiment ideas?
    - What models do you use?
    - How do you know if an experiment idea is good?
    - What fraction of your experiment ideas are good?
3. **Transition from development to production**
    - What process do you follow for promoting a model from the development phase to production?
    - What do you look for in code reviews?
    - What automated tests run at this time (between development and production)?
4. **Validation datasets**
    - How did you come up with the dataset to evaluate the model on?
    - Do the validation datasets ever change?
    - Does every engineer working on this ML task use the same validation datasets?
5. **Monitoring**
    - Do you track the performance of your model?
    - If so, when and how do you refresh the metrics?
    - What information do you log?
    - Do you record provenance?
    - How do you learn of an ML-related bug?
6. **Response**
    - What historical records (e.g. training code, training set) do you inspect in the debugging process?
    - What organizational processes do you have for responding to ML-related bugs? Do you make tickets (e.g. Jira) for these bugs?
    - How do you react to these bugs?
    - When do you decide to retrain the model?

APPENDIX B: Transcript

What company are you working for?

Nowadays I work as a freelance engineer. So, I have a contract with an organization, and I work with them on a contract basis. I have my own business and I do freelance assignments like this. I'm still in the trenches working quite technically, really technically actually, and educating I would say. Now it is within machine learning, or AI if you like to call it that.

Ok, so are you currently working on a machine learning task or model?

Yes

Specifically, if you could tell us what it is about?

Since I'm bound by contract I can talk a little bit about it, give an overview without any details.

Yes we understand. That would be great

If we could stick to a general level, we could stick to that and hopefully it will be useful, I will do my best to explain and share what I know. But ok, indeed, I saw the questions you sent me. And in the beginning one of the questions were "do you work on classification or regression models"... and ok, what's really the difference? I can argue like that I use TensorFlow and PyTorch, mostly PyTorch, and I train and I generate the data and help my customers to clean and get hold of data, and organize the data, and then we train models. I you use PyTorch that means some type of deep neural network, could be classification, could be other things. I do a lot of object detection in fact, and also a little bit of object character recognition, OCR. So basically, the raw data is video files. We have access to plenty of footage, hours and hours from the field. And then the task is to help my customers to make sense of that data. Mosly, we do object detection, and then my comment is this regression or classification, I mean in the end if you look at classification I would argue, isn't that also regression. In some sense you are minimizing some kind of cost function.

The models and tasks you are working on, are you part of trying to come up with what's next or do they just assign you what they want you to create?

No, no really I'm working more on an expert senior level, and also now it depends on the maturity of the organization you are working with in terms of machine learning knowledge. And the organization I work for do not know a lot of machine learning, even though they generate a lot of data and use machine learning to help out analyzing and understand the data, in this case I could more in a way take part in driving and setting up a direction on where to go.

So when trying to find an idea, what's the process, how do you know what's the right route to take?

You never know, there's a certain amount of R&D, if you do new things in a sense. A new application, the data is new you have to try things, try it, evaluate it, and compare it with other methods and then see where it goes. It's always like that, I mean it is engineering, it is never a straight path, you go right and left.

Do you often have ideas that turn out to not be good, do you often "fail"

Sometimes, I wouldn't say often, but sometimes. You should allow yourself to fail sometimes.

That's reasonable. Then, when you have developed a model, what's the process of taking it from an experimental- and development stage to actual production and deployment.

I have little experience of that. Where I work now is not a production environment. It is more of a research environment. We do more prototyping, helping my customers, I help them to generate tools that they can benefit from during their research. When you say production, I immediately think about, for example, the car industry. If you go into production there, then it is safety critical and it becomes a whole lot more strict. I just have some experience from it so when you are in an experimental phase my first comment, and I think a lot of people think like that, you use Python. Python is nice and it is easy to put together something, to build something and then learn from that. I'm not sure, but when you go to production phase then you probably you leave python. You translate your models and you spend a lot of time actually, it depends on how critical the processes are, rewriting. But all the code has to follow a certain standard and be tested meticulously. So you write in C or C++ or something like that, that is what I'm used to. I think that costs a lot of money for companies going into production, of course thinking about scaling up and mass markets and such things. Another thing is when you go from prototype to production, perhaps you have to scale down your model to fit the device, so usually when you develop you use a competent server or your desktop computer and you have access to GPUs and stuff like that but if you go into the target application sometimes, probably, it could be an embedded device, and on that you might not have as much computing power, mostly. Although you have some embedded devices from Nvidia and other vendors, but still it is limited. Another aspect: you leave training, and you have a model, and you only do inference. In production you do inference on the target device. So, then you optimize the model to run well on the target device. So, you have to do quantization, leave float point precision and go into integer precision or something like that and the speed up and... yeah I guess you know all this.

Some of it haha. Yeah

So this is what I think when you say "go into production". What else... some people...it is not so common that you think about training in a production environment. There are, have you read about decentralized learning or federated learning?

I don't recognize that.

That is something you could look up, federated learning. I think it was coined by Google five years ago or something like that. I think there's a blog post from Google, or even some research paper. So basically, instead of having data and computer power on a central spot, you break up and move out the data and move it out to the edge where you have the sensors and small embedded devices. So you do the training on the small devices and then they communicate in some way, either through a central device or more peer to peer. And then you could start talking about doing training on the actual device. It is kind of nice. And I think this is what Google and Apple and these companies do already on your mobile device.

Alright. But then, going back to something you mentioned in the beginning. The performance measure for an experiment setting compared to a production setting. What is it you look for, what is you performance measures.

What is come to... there is always a measure.

Not specifically the metric, but are you looking more specifically on how good the code is structured, or time efficiency.

When I do prototyping, I'm not working in a big team. I try to keep it tidy and nice, and organize and generalize it to some extent. Well that is another thing if you go into production perhaps you are part of a team and you work in a more continuous integration kind of fashion, where you have code review and standards to follow, nighty builds, unit testing and where you set up KPIs to do testing on different kind of hardware. You have a whole machinery running and it is quite elaborate to set up, but I guess if you work in a professional environment. Have you seen Grafana, it is a dashboard application. It is kind of nice so you hook up your data to Grafana and then you visualize everything on a dashboard. So, every

morning you can go in an see how the nightly build went and what was the commit that broke the code, and the you could go back and blame someone, you know.

Are you currently using that

No no. Nothing like that at the moment

So in a smaller team, what are you using?

It is very elementary now. It is not like that at all. Basically it just comes down to keeping track of things on git and basically building up things like that now for my customer a little bit. Git is quite standard, straight forward to version control code. It is very powerful and established. But now when you start working with machine learning, if you want to do that in a structured way, you will probably have to version control your data, you need to version control your models. To keep track of changes and have a way if managing the how the development goes. So new data, always comes in I guess .

Are you mainly using Python for your data collection and formatting?

Yeah for sure, I use mostly Python nowadays.

Are you responsible for choosing what kind of data you are going to use?

Well not responsible, well to some extent I can influence how the raw data is collected but most data is already collected. I don't know, it is quite a lot of data, not for everyone, but I work with data in the range of 10 TB of data, and it is footage, it is video. Yes I can influence. What we do with the footage and the video, the thing is from that we generate annotated images so you have to pick images in the footage and go in there and draw rectangular boxes and label them. That is actually something we do right now.

Do you ever change the data set you are using.

Yes. Right now, in one of the projects we are building a model from scratch basically and from the beginning we didn't have any annotated data at all, so we started building a small set of data, and then we iterate on that several times to intruse (???)data. It is not formally what you call active learning, but it is related to active learning.

M**oving on from datasets, Walter do you have something about data you want to ask. No I think we touched on a lot, of course we are interested to know how you transfer these large amounts of data to co-workers or generate statistics of the data?**

Well, good question. That is something we are building, I think. In terms of generating statistics on the data, I can browse through the data with scripts and generate reports, but it is not any formal things. I can't recommend you any tools for that. As for sharing, I mean we work in Sweden, we are moving data across the country from coast to coast basically. And when it comes to 10 TB of data, at least our network bandwidth is not so high so if you transfer it on the network you have to wait a long time so we actually physically transfer the data on hard drives.

Ok, that's is interesting to hear.

And then when you get it on the spot you have networks drives that are connected to desktops, and I think people that analyze footage can have access whenever they are on the network.

Sure.

But that's fascinating. The amount of data is increasing and moving it is not so easy.

Alright, then we could move on to monitoring, because we are done with the development phase. Both when you develop a model, but also when you feel

finished, how do you track the performance of a model and evaluate its performance?

Well, I can tell you from experience of what I've seen. Like I was saying, if you work in an industrial environment you already have continuous integration in place. That is quite important work, to sit down and define your metrics, you have to define your tests. You have to have a set of test data that is not touched by the training at all what so ever, and then you run your models. You calculate the KPIs , you store them over time and you plot a graph and then you se the trends.

So you are continuously tracking the performance of the model as you develop it?

I'm not doing it now. But I've been in environments where we did this and this is how it works.

When you decide upon your metrics, how is that process like? Are you starting out with an educated guess and then evaluating?

I guess, that is a very general question what is happening, you have an assignment to come up with your KPI, a bunch of people might have it, it will be kind of a task. You have to think about it, study literature see what other people have done. And then you come up with a proposal and you discuss it in a meeting with the product manager and other managers and there is a decision to go with it. And then it is implemented, maybe it is discovered that it isn't working so well and then it has to be changed.

Ok, so what's your process like when you have bugs in your system?

Where I work now, when I have bugs I honestly don't track them. I go in and fix them. Professionally, or more in a team, it should be a tracking system in place and that goes hand in hand with the nightly builds and if some tests have failed you have to assign someone to look into it and see if it is a bug. If you work in a team, you have the main branch, you check out your own branch, you have your ticket and do your work. Before you are even allowed to commit usually you run your own branch on a set of tests and hopefully that will catch any bugs you have. The worst thing is if you have bugs coming from a customer. You have shipped something, and a customer is calling you telling you something is wrong then that is critical and of course you have to follow it up.

Do you run any automated tests or processes on your code?

No, I don't have it right now?

Are you doing all your testing manually?

Yes, I test manually, it is such a small thing. It is not really a production thing. I don't have the time and resources to work like that. It is good but you have to find a balance. I work on such a small project, and I haven't implemented automated tests and so on yet.

So, you are currently spending most of your time coding to models, looking for datasets or…?

Yes, exactly that is what I do, help generating data and training the first models to see if they get an understanding on how they perform. But I think in the long run, even the customers I work with right now and the projects I work on , they will probably have to go in that direction for sure. You have probably been out programming a little bit yourselves.

Yes.

Everyone is told that you should do test-based development but not a whole lot of people are following it strictly. But sometimes I do, it is kind of nice. It is good to have tests as some kind of back up.

I was going to ask, what is something that happens that makes you revisit your model or retrain your smaller model that you were talking about. Where do you decide to stop the research and start from the ground up?

One reason is if I have more data. Another thing is if there are unbalances in the data and I have more data from certain classes that will balance the data then I retrain. So I think it is verry driven by data. When you retrain it is not driven by a new algorithm coming up, it is more about data. Data is so important, and without good data you are lost.

Do you use any models that are publicly available and then train on top of them or do you start from the ground?

Sometimes I start from the ground. I don't work on classifying several hundred objects. Sometimes I have two classes, and data. People usually start with Resnet. Resnet is a standard classification network, it is quite old now, but it is good. And there are different sizes. So people take one of these, and they take a standard image 255x255, image, and they run it in this Resnet, I don't know what number, there are different numbers, and it has 20 million degrees of freedom and they run it and they start with some pretrained weights and they just run it and they are happy. What I've done is take a Resnet model and downscale it. I did some modification in it and make it much smaller, so we reduce the degrees of freedom by a order of magnitude, and it did very well. Very fast. So I think, sometimes I, this is just personal reflection, but I am so amazed. These models and algorithms that are out there they seem to be very good and quite robust, but I'm so amazed that when you do linear optimization with 10 million degrees of freedom, what can you possibly know about the characteristics about such a solution, I mean it is completely crazy. Therefore I like to downscale the degrees of freedom. Not that I will be any wiser because of that, but at least it will not overfit so much. But that is one thing. It takes a whole lot of time, but it is fun. In projects I've had over the years, with student and so on, I used YOLO5 quite a lot. DO you know YOLO5?

No

So YOLO5 is object detection. There's a whole range of YOLO object detection algorithms, from YOLO 1,2,3,4 and there is 5. Some even claims there is six of them but it is kind of starting to degenerate a bit but, Up to version 4 it was based on a kind of dark net. An architecture that was implemented as kind of the backbone, that is called dark net. But then a guy started to implement everything from scratch in Pytorch and YOLO5 was invented. It's quite an active GitHub repo. They iterate on in all the time and update it. That's why I like it, it is kind of active and makes code good and robust and you can google if you have questions you know there is forums and so on. So I used YOLO5 quite a lot actually and it is doing a pretty good job actually.

But if you work in a production environment you probably have to do everything from scratch in a C++ environment or something like that.

Alright.

Is it kind of outdated to talk about c++ with you guys or do you use Go and Rust and these kinds of languages?

Walter: We were just talking about this, for our classes we usually use Python a lot for how easy it is for students to learn, but then when it comes to lower level programming we still use C and C++ and so on

Vera: That's also maybe a difference, because in Sweden I've never used Python. In Sweden we use C, C#, and C++ and Java mostly. Never Rust though

Walter: Yeah Vera and I were laughing because I mentioned how I wanted to learn Rust right before this meeting. We were talking about it.

Ah ok.

I think I'm happy with the questions. We got a lot of content, that is good. Thank you for your time…(Finishing up)