# Machine Learning in Production: A Literature Review

Yizhen Zhao
12913405(UvA) 2658811(VU)

March 17, 2021

**Abstract**

Machine learning and AI are becoming indispensable in data-driven businesses. While running and maintaining machine learning in production can be challenging. Over the years the adoption of DevOps principles in software engineering enables developers to deliver their products in an efficient and scalable way. Now, the same trend is shown in machine learning projects, where the developers try to adopt DevOps principles to machine learning, so-called MLOps. This paper evaluates the current challenges in running ML in production by reviewing and analyzing relevant academic literature, along with other two relevant topics about the importance of MLOps and how to apply the DevOps principles to machine learning projects. The paper concludes with summarized answers from the literature study to the three research questions.

## 1 Introduction

Throughout history, machine learning (ML) becomes more and more popular in business, it offers a powerful toolkit for helping solving complex real-word problems. As we are living in a world surrounded by massive data, we are trying to take advantage of those data and turn it into action. There are two main techniques in machine learning, supervised learning and unsupervised learning[1]. Supervised learning is when you train the machine with labeled data. Labeled data means the data is already tagged with the correct answer, therefore, helps you to predict results for unforeseen data based on the previous experience. While unsupervised learning works in a different way, it does not need the labeled data. Instead, the model itself works in its way to discover unknown patterns in data.

Nowadays, researchers and enterprises can make use of machine learning to discover meaningful information and patterns hidden in the massive data for their business. For example, COVID-19[1] influences the whole world since 2019

[1] https://en.wikipedia.org/wiki/COVID-19_pandemic

and lots of researchers devoted to use machine learning to help the world to predict the trend of the epidemic, projects like predicting the diagnosis based on the symptoms[2] so that it can mitigate the burden of the healthcare system. Enterprises like Airbnb[3] uses machine learning to help them find out the hidden patterns behind the host's and guest's behavior. They record each search query made by guests on Airbnb search engine and try to discover what affects the host's decisions to accept accommodation requests and how to increase acceptances and matches on the platform. This helps both hosts and guests to find their preferences. Machine learning in Airbnb not only helps them increase their business value but also improves the user experience.

Just like DevOps, which is aiming to increase an organization's ability to deliver software applications at high velocity, evolving and improving products at a faster pace. MLOps, which is DevOps for machine learning, acts in the same role in running ML in production. It aims to automate the entire ML lifecycle, increases the pace of model development and deploying and helps deliver the innovation faster.

Sculley et al.[4] mentioned in their paper that in the read-world machine learning system, it is relatively fast and cheap to develop, but difficult and expensive to maintain on the long-term. The main objective of this paper is to conduct a literature review of the machine learning system in production, where we intend to find out the challenges and difficulties in building and maintaining ML system in production environment, and why MLOps has been introduced into machine learning to help to scale the ML system in production. What are the characteristics and principles for it? Thus, we identify the core concept by analyzing existing literature to form a deep understanding of running ML in production.

I organize this paper into five sections. First, I introduce background information in the ***Introduction***. Then in ***Methodology*** I describe the research questions and motivations for choosing them. In the following, in ***Discussion of Literature***, I classify the literature into three sections to answer the research questions. The summary of this paper is included in ***Conclusion***, and then ***Reflection and Limitation*** is included. Last but not the least, terminologies mentioned in this paper are explained in `Appendix`.

## 2 Methodology

The topic for this literature review is "Machine learning in production", which is running and deploying machine learning in production, it is different from developing machine learning for research. Therefore, the research questions and the rational motive for the stated research questions are:

- Q1: What are the current challenges in developing machine learning projects?
  - I try to identify the challenges researchers and teams encountered when implementing machine learning projects.

- Q2: Why MLOps? What is the difference between DevOps and MLOps?
  - I try to identify why we need MLOps to help us implement ML in production, why DevOps cannot satisfy the requirements of MLOps and what are the characteristics of MLOps.
- Q3: How to apply DevOps principles to MLOps?
  - I try to identify what kind of principles MLOps can learn from DevOps.

In the second step, to answer the research questions, I selected relevant literature for six scientific publications and three credible online articles. For published papers, I used searching tools, such as Google Scholar[2] and Mendeley[3], to select papers that have a relatively large number of citations. For online articles, they are related to the keywords like "MLOps" or "machine learning" and then I selected those that were highly relevant to my literature topics, such as the why we need MLOps, the importance of it and etc.

In the third step, based on the research direction, I classify and group those scientific papers and online websites into several topics, which include ***Definitions***, ***Challenges in machine learning projects***, ***Importance and difference of MLOps in production*** and ***Applying DevOps principle to MLOps***. Finally, I present a conclusion, reflection and limitation of this paper.

# 3 Discussion of Literature

## 3.1 Definitions

As the topic of this literature study is "Machine learning in production", we might wonder what is machine learning in production? What are the differences between machine learning in academic research and in practice?

Obviously, there are some key differences when dealing with ML in academics and production. According to the articles[5][6], the main differences between them are, first, they focus on different aspects. Machine learning in research is more about *science* and *theory*, a ML researcher tries to improve the precision of a model, maybe focus on a specific aspect, or implement models that solve custom scientific problems. However, for an industrial-based ML project, it is more about *engineering*. It is not only about implementing ML models but also many other engineering works, such as integrating ML system into existing industrial settings, delivering Software as a Service (SaaS), applying DevOps principles so that it can be used in scale and continuous delivery in production and etc. Second, they start from different perspectives. Academic research is often driven by competitions or challenges with a cleaned and almost structured dataset. Computational cost is not their main concern, they are more careful about the accuracy of the ML models. While in practical, real-world data is

[2]https://scholar.google.com/
[3]https://www.mendeley.com/search/

always more complex and unstructured, therefore, the ML development and deployment effort in industrial may not be only on the core of ML algorithms, but on data cleaning, formatting and deliver a product as fast as they can.

Scrulley[4] mentioned in their paper it might be surprising to know that compare to ML in academics, only a small fraction of the code in ML systems is actually devoted to production, shown in Figure 1. It is true that in real-word ML systems, the ML code will be surrounded by several different components.

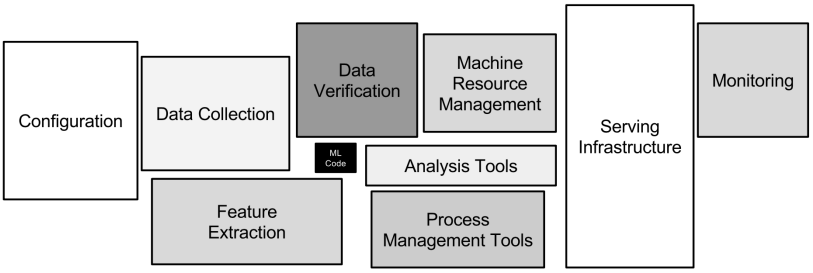


Figure 1: Only a small fraction of real-word ML system is composed of ML code (the small black box). Source: *Hidden Technical Debt in Machine Learning Systems*[4].

## 3.2 Challenges in machine learning projects

As machine learning systems are not the same as the traditional software system, many other components involved in ML systems, therefore, several challenges in managing and maintaining ML system have emerged.

Sculley[4] claimed the long-term costs in ML systems. I summarize it into the following aspects:

- Management debt

The typical software management method can handle the code level, but not the ML-specific system level. There is an extra set of ML-specific issues. In the production environment ML system, usually, the goal is not to run and maintain a single model, but dozens or hundreds of models are running simultaneously. This might lead to management debt. For example, how to update and assign the power configuration for models?, how to keep track of experiment results?, and how to monitor the whole production pipeline?.

- Data dependencies are more costly than code dependencies

Data used in the ML system might be unstable which means it can change over time, as it reflects the real-world situation. Different input of data can lead to different behaviors, which makes us lose control of our ML system. The importance of versioning data can be seen here as it might be one of the strategies to deal with unstable data, to create a frozen version of the data so that we can trace back if anything unexpected happened. While versioning also carries

its own cost. For instance, the cost of maintaining multiple versions over time. Moreover, because of the unstable data, proper data testing should be involved, to make sure the whole system's functionality.

- Complex modular boundaries

Traditional software engineering practices utilize modular design to help maintain the whole system, so you can change one part without interfering with others. While for the ML system, it is difficult to set such clear boundaries. For example, a ML model depends on several features, if one feature has been changed, namely, the distribution, the importance, it might influence the remaining features.

- Configuration debt

The ML systems in production environment contain many components. A system that incorporates ML might end up with high-debt configuration debt. Configuration like which features to use, how to prepare data, the possibility to include pre-processing and post-processing of data, how the feedback loop looks like and etc. Any mistakes that happen in configuration can be costly, leading to loss of time, more labor work needed, paying more budget for computing resources, or productize issues.

Schelter[7] shared a similar point of view on data management challenges. To automate and accelerate ML lifecycle management, we need to understand and keep track of the training and validation data used as well as hyperparameters, accuracy results of the model. A centralized place for storing ML artefacts and keeping track of experiment results is needed. Another challenge in model management is the multi-language codebase. As we mentioned before in Sculley's paper[4], ML systems usually contain several different components. And it could be the case that they use different programming languages for different components. For example, Spark is more preferable for dealing with the large scale of dataset, and many popular libraries for ML are written in python. Because of this characteristic, they might need different configurations. How to exchange data between components efficiently and reliably is also a challenging task.

Since ML can only learn what it is taught, a crucial prerequisite for using ML is data variability and quality. Ansreas in their paper[8] described the similar data challenges in ML used in production. It is not surprising that data preparation is the most time-consuming part of a ML project. Provide the necessary data, validating and cleaning directly related to the quality of the model.

### 3.3 Importance and difference of MLOps in production

MLOps, which is DevOps for machine learning. The traditional way of handling software systems in production is by applying DevOps principles. While in the past, software engineering was inefficient and releases took months to ship without DevOps. Now thanks to DevOps, continuous integration (CI) and continuous delivery (CD), the releases become faster and more efficient. So

why is it different for using ML in production versus software engineering? Is it a crucial requirement to introduce MLOps in the ML system in production? The answer is yes if you want to use ML in real-word with reproducibility and reliability.

In this article[9], it mentioned a fundamental reason behind is the ML lifecycle is iterative. You might find or fix issues at each of the steps in the lifecycle (e.g. data preparation, model building, deployment in production and etc), and hence, necessitate modifying other stages in the lifecycle. It is not a static end-to-end workflow, what is needed is a process in which each component, the data, code, model, results and each experiment are automatically tracked and versioned so that the whole system is under control, you can easily fix changes and reproduce any stages in the lifecycle. Now ML is in a similar situation to software engineering in the past, without DevOps. The whole ML lifecycle and releases are inefficient, and a lot of manual work is involved. That is a reason why MLOps is needed, by bringing similar principles of DevOps to ML systems.

Yuri[10] pointed out the demand for continuous development, continuous integration and continuous delivery motivates enterprises to adopt DevOps principles. It increases the efficiency of their development and delivery. The same demand is needed for ML systems. The ML system and traditional software system, they have general similarity, ML systems can also benefit from the DevOps principle, for example, by adopting version control systems like Git[4]. At the same time, they are different in certain processes (e.g. data preparation, data cleaning).

The ML infrastructure is complex and different from the original software engineering[9]. ML workflow includes several new parts like datasets, models, models hyperparameters, metrics and etc. The traditional DevOps cannot satisfy some requirements in ML systems. According to these new parts in ML, not only the code needs to be tracked and versioned, but also the datasets, models, metrics and etc.

Yuri[10] introduces the difference between DevOps and MLOps as that, DevOps helps data engineers continuously build, maintain and improve the entire service lifecycle. While MLOps is mainly for data science. Data science is about developing models, in this case, ML models, that are used in production. And one characteristic of this model development is the ML process includes iterative model development, experiment and continuous improvement even after the product is delivered to production. MLOps is helping data science speed up their work process with data in a scalable and reliable way.

### 3.4 Applying DevOps principle to MLOps

DevOps[5] is a set of practices and tools based on software engineering, in order to help organizations to improve their ability in delivering applications of services faster than the traditional software processes. On the other hand, Agile[6] is

[4]https://git-scm.com
[5]https://www.synopsys.com/glossary/what-is-devops.html
[6]https://www.atlassian.com/agile

an iterative approach for software management. DevOps and Agile are the backbones of supporting teams to build an environment that can continuously develop, improve and deliver their applications and services. While MLOps is just another domain where some of the DevOps and Agile principles can also apply.

Yuri[10] introduced how to apply DevOps to data science projects on a general level. When considering the research methods and research cycle, they have a general similarity. Steps like propose hypothesis, experiment design, data preparation, data analysis, model building, evaluation, monitoring and improvement. All steps should be documented, stored, linked and to ensure reproducibility. The continuous development, improvement and delivery can be applied to data science workflow.

Karamitsos pointed out in their paper[11], the main idea for applying DevOps principles to ML system is to adopt CI/CD practices. This means we wish to automate and monitor all steps in ML system, including integration, testing, releasing and deploying in production. In the ML system, CI uses not only for testing and validating code but also for data and models. CD is no longer about a single package or service, instead, it needs to handle the ML pipeline and automatically deploy ML services. They proposed a MLOps framework that mainly focuses on continuous training and testing of the model. They used the general ML lifecycle and the highlights of the entire process are in CI and CD stages. In the CI stage, they build ML-related code, data-related tasks (e.g. preparation, analysis, cleaning), run several experiments to get the trained model and other ML artefacts, testing and validating models and finally package the ML artefacts. Then the outputs of this stage are to be deployed in different environments (e.g. staging, production) of CD. In the CD stage, we deploy ML artefacts created in CI stage to staging/production environments. The purpose of CD is the delivery of ML models.

Karlas[12] mentioned the same DevOps principles, continuous integration and continuous delivery, that can apply to ML projects. CI services help engineers deal with software development cycles, start from the building, testing, and deploying applications in an iterative and automated way. ML applications are not much different in this process, it repeats this iterative process until developers are satisfied with the model quality. They proposed a CI framework that includes steps like develop, build, test and release. First in *develop*, engineers working on writing code for pre-processing data, tune parameters and all ML artefacts are handled in this step. Then in *build*, it automatically triggers the build process which starts training the model. In *test* process follows the build process, returning the valuation result of models to developers for further improvements. If all tests passed and developers are satisfied with the end result of the model, it starts the *release* process. The tricky thing in CI for ML is that the testing result is not like the regular software test result which is deterministic. Testing results for ML projects might differ when the dataset used changes.

# 4 Conclusions

DevOps took a few years to become an accepted practice in software engineering. In the early year of applying DevOps principles, there were many challenges as well. Now, the same trend has started in the data science and data engineer domain, which is called MLOps. As machine learning systems are not the same as the traditional software system, challenges like how to manage the models in production can be even harder. MLOps might also take a few years to become a mature practice that can be accepted by the marketplace. Although it still has a long way to go, a mature and successful MLOps strategy can make collaboration and integration easier, make ML lifecycle and deploy ML in production more efficiently.

This section presents the summarized answers to the previously identified research questions that have been drawn from the literature study.

- Q1: What are the current challenges in developing machine learning projects?

There are two main challenges in building ML in production, which are on the management level and dependency level. At the management level, as ML has different components, different parts might need different configurations or tools, and managing the whole ML lifecycle that each part of it might happen in different environments, can be a challenging task. For the dependency level, one of the biggest dependencies is data. Data used in ML can be unstable and change over time as it reflects reality. Training and validation data used is also crucial in ML projects, an error that happens in data could lead to different behaviors. Data preparation is indeed a time-consuming task, but the quality of data needs to be guaranteed. To use reliable data in ML projects, data versioning is important to introduce. While versioning for data can be challenging as it is not the same as versioning code.

- Q2: Why MLOps? What is the difference between DevOps and MLOps?

MLOps helps data scientists to manage the ML lifecycle in production in a reproducible and reliable way. ML projects are an iterative process, start from data preparation to the final model evaluation. Data scientists might repeat the process several times until they are satisfied with the result. The biggest difference between DevOps and MLOps is that the ML system has a more complex infrastructure than the traditional software system. DevOps cannot satisfy certain ML-related requirements. MLOps helps data scientists continuously develop, experiment and improve models and at the same time, keep track of the dataset, models, metrics, etc.

- Q3: How to apply DevOps principles to MLOps?

Although ML systems are different from traditional software systems in some parts. They do share certain similarities. ML system can also benefit from

continuous integration (CI) and continuous delivery (CD). The whole ML lifecycle can be automated and monitored. CI pipeline mainly focuses on ML model development process, from preparing the data, tuning parameters, running experiments, building and validating models. CD pipeline is responsible for deploying ML artefacts coming from the CI pipeline into production. While during model development, the tools, mechanisms or practices used for managing datasets, experiment results, metrics and etc, might differ depending on the developer's requirements.

# 5 Reflection and Limitations

During the research and practicing machine learning in production. There are some challenges I have already seen that also be identified by the literature. The biggest one is the data used in ML projects. The process of creating an unbiased training set for ML projects is quite time-consuming and error-prone. It depends on manual labeling and creation. The labeling result might have errors as we have a different understanding of the definition of the tasks that we are doing, therefore, lead to the different behavior of models. Dataset is usually biased, while sometimes the manual created unbiased data cannot really reflect reality. The other challenges on system-level, they cannot be seen immediately but worth keeping that in mind when developing ML projects.

A limitation of this literature study is that the limited academic papers related to MLOps. Topics like deploying ML in production, applying the DevOps principle to ML, the importance and architecture of MLOps in industrial settings and etc, are quite new topics in the most recent years. Although there are lots of technical blogs, like Azure MLOps[7], introducing their approaches and MLOps architecture, only limited academic papers researching or describing the same thing. As the relevant research of running ML in production still needs time to investigate and develop, but fortunately, there is still some relevant literature that can be used.

[7] https://docs.microsoft.com/en-us/azure/architecture/example-scenario/mlops/mlops-technical-paper

# References


[1] "Supervised vs unsupervised learning: Key differences. https://www.guru99.com/supervised-vs-unsupervised-learning.html."

[2] Y. Zoabi, S. Deri-Rozov, and N. Shomron, "Machine learning-based prediction of covid-19 diagnosis based on symptoms," *npj Digital Medicine*, vol. 4, Dec. 2021.

[3] "How airbnb uses machine learning to detect host preferences. https://medium.com/airbnb-engineering/how-airbnb-uses-machine-learning-to-detect-host-preferences-18ce07150fa3."

[4] D. Sculley, G. Holt, D. Golovin, E. Davydov, T. Phillips, D. Ebner, V. Chaudhary, M. Young, J.-F. Crespo, and D. Dennison, "Hidden technical debt in machine learning systems," in *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'15, (Cambridge, MA, USA), p. 2503–2511, MIT Press, 2015.

[5] "Machine learning in academic research v.s. practical. https://towardsdatascience.com/machine-learning-in-academic-research-vs-practical-5e7b3642fc06."

[6] "Don't make this big machine learning mistake: research vs application. https://towardsdatascience.com/dont-make-this-big-machine-learning-mistake-research-vs-application-bd52d5a9a8b9."

[7] S. Schelter, F. Bießmann, T. Januschowski, D. Salinas, S. Seufert, and G. Szarvas, "On challenges in machine learning model management," *IEEE Data Eng. Bull.*, vol. 41, pp. 5–15, 2018.

[8] A. Mayr, D. Kißkalt, M. Meiners, B. Lutz, F. Schäfer, R. Seidel, A. Selmaier, J. Fuchs, M. Metzner, A. Blank, and J. Franke, "Machine learning in production – potentials, challenges and exemplary applications," *Procedia CIRP*, vol. 86, pp. 49–54, 2019. 7th CIRP Global Web Conference – Towards shifted production value stream patterns through inference of data, models, and technology (CIRPe 2019).

[9] "Why do data scientists need devops for machine learning (mlops)?. https://dotscience.com/blog/2019-10-21-devops-for-ml/."

[10] Y. Demchenko, "From devops to dataops: Cloud based software development and deployment,"

[11] I. Karamitsos, S. Albarhami, and C. Apostolopoulos, "Applying devops practices of continuous automation for machine learning," *Information*, vol. 11, no. 7, 2020.

[12] B. Karlaš, M. Interlandi, C. Renggli, W. Wu, C. Zhang, D. Mukunthu Iyappan Babu, J. Edwards, C. Lauren, A. Xu, and M. Weimer, "Building continuous integration services for machine learning," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery amp; Data Mining*, KDD '20, (New York, NY, USA), p. 2407–2415, Association for Computing Machinery, 2020.

# A Glossary

**CD** Continuous delivery or continuous deployment. 7

**CI** Continuous integration. 7

**CI/CD** CI/CD generally refers to the combined practices of continuous integration and either continuous delivery or continuous deployment. 7

**DevOps** DevOps is the combination of cultural philosophies, practices, and tools that increases an organization's ability to deliver applications and services at high velocity. 2, 3, 5–8

**ML** Machine learning. 3–9