https://christophergs.com

# Best Practices for Building Machine Learning Systems

*Key Papers Brief Summary*

---

## Contents

https://christophergs.com

## Introduction

Developing and maintaining machine learning systems is hard. The technical debt easily incurred by such systems is well explained in an older paper from Google "Hidden Technical Debt in Machine Learning Systems". Despite this complexity, clearly defined best practices are in short supply. Why is this?
Whilst academic machine learning may have its roots in research from the 1980s, the practical use of machine learning systems in production is still relatively new. With a few pioneering exceptions, most tech companies have only been doing ML/AI at scale for a few years, and many are only just beginning the long journey. This means that:

- The challenges are often misunderstood or completely overlooked
- The frameworks and tooling are rapidly changing (both for data science and data management)
- The best practices are often grey
- The regulatory requirements are changing all the time (think GDPR)

This is the situation. And that is why any research papers detailing best practices around system design, processes, testing and evaluation written by companies with experience in large-scale ML deployments are extremely valuable. Drawing out common themes and issues can save you and your company huge amounts of blood, sweat and tears. Two of the most experienced companies in the ML space, Google and Microsoft, have published papers in this area with a view to helping others facing similar challenges. The two papers are:

1. The ML Test Score: A Rubric for ML Production Readiness and Technical Debt Reduction (2017) Breck *et al*. (Google)
2. Software Engineering for Machine Learning: A Case Study (2019) Amershi et al. (Microsoft)

In this brief report, I intend to draw out the key learning points from these two papers, discuss their implications, and boil down the lessons for readers.

---

## The Research Angles / Methodology

The Google paper focuses more on ML system testing and monitoring strategies that can be be employed to improve such systems in terms of reliability, reducing technical debt and lowering the burden of long-term maintenance. The Google paper is structured around 28 tests, a rubric for gauging how ready a given machine learning system is for production. These tests are “drawn from experience with a wide range of production ML systems”.

The Microsoft (MS) paper takes a broader view, looking at best practices around integrating AI capabilities into software. The paper presents the results from surveying some 500 engineers, data scientists and researchers at Microsoft who are involved in creating and deploying ML systems, and providing insights on the challenges identified.

Both papers offer a lot of value. Whilst the Google rubric can be more easily used to assess a given system, the MS paper offers insights on how a team’s focus may need to shift depending on experience level, which is a useful additional layer of granularity. Let’s have a look…

---

## The Challenges of Machine Learning Systems

The two papers agree that the challenges of ML systems exceed those of traditional software systems, and both concur that the root cause of this is the dependence on uncertain data and models.

> "*ML system testing is also more complex a challenge than testing manually coded systems, due to the fact that ML system behavior depends strongly on data and models that cannot be strongly specified a priori*"
>
> - *Breck et al., 2017*
>
> *"The amount of effort and rigor it takes to discover, source, manage, and version data is inherently more complex and different than doing the same with software code. Second, building for customizability and extensibility of models require teams to not only have software engineering skills but almost always require deep enough knowledge of machine learning to build, evaluate, and tune models"*
>
> - Amershi *et al*., 2019

Linked to this challenge, both papers highlight that processes and standards for applying traditional software development techniques, such as testing, and generally operationalizing the stages of an ML system are rare:

> *"Software testing is well studied, as is machine learning, but their intersection has been less well explored in the literature."*
>
> - Breck *et al*., 2017

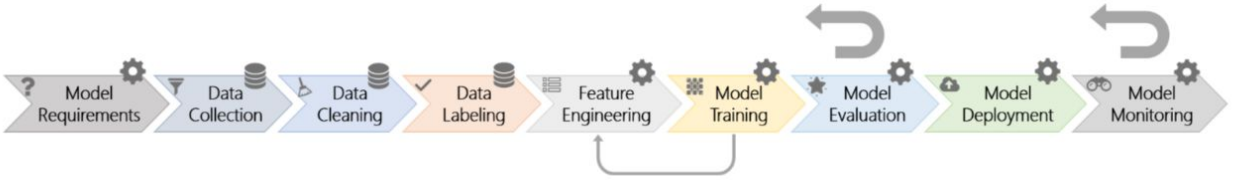
In the MS paper, the authors highlight that "In the last five years, there have been multiple efforts in the industry to automate this process by building frameworks and environments to support the ML workflow and its experimental nature […] However, ongoing research and surveys show that engineers still struggle to operationalize and standardize working processes." These challenges represent the starting point for the recommendations.

---

## Common Themes and Best Practices

### End-to-end pipelines

*Figure 1: The MS machine learning workflow*




In the Microsoft paper, end-to-end pipeline support is highlighted as crucial, particularly automation of all stages of the ML workflow (*Figure 1* above), including training and deployment. The importance of reproducibility across the pipeline is evident from the model recommendations around versioning:

- Each model is tagged with a provenance tag that explains with which data it has been trained on and which version of the model.
- Each dataset is tagged with information about where it originated from and which version of the code was used to extract it (and any related features).

This same emphasis on the overall pipeline and reproducibility is mirrored in a range of the tests from the Google paper rubric, notably:

- Training is reproducible (infra test 1)
- Every model specification undergoes a code review and is checked in to a repository (model development test 1)
- The full ML pipeline is integration tested (infra test 3)
- Models can be quickly and safely rolled back to a previous serving version (infra test 7)

### Data Availability and Management

The MS paper, since it is focussing on broader best practices, looks at the importance of managing data availability, accessibility and quality.
In addition to availability, our respondents focus most heavily on supporting the following data attributes: “accessibility, accuracy, authoritativeness, freshness, latency, structuredness, ontological typing, connectedness, and semantic joinability.” (Amershi *et al.*, 2019)
Some of the Google paper tests are obviously geared to check against these types of concerns. This can be seen in:

- Feature expectations are captured in a schema (data test 1)
- All input feature code is tested (data test 7)

### Managing the Gap Between Research and Production

The MS paper highlights that:

> "*ML-centric software goes through frequent revisions initiated by model changes, parameter tuning, and data updates. A number of teams have found it important to employ rigorous and agile techniques to evaluate their experiments."*

This same concern about model updating is reflected in particular by the infrastructure tests advocated by Google:

- Model specification code is unit tested (infra test 2)
- Model quality is validated before attempting to serve it (infra test 4)
- Models are tested via a canary process before they enter production serving environments (infra test 5)
- Data invariants hold in training and serving inputs, i.e. monitor Training/Serving Skew (monitor test 2)
- Monitor that the model has not experienced a regression in prediction quality on served data (monitor test 7) - also linked to data concerns.

## Compliance

Both papers bring up compliance and privacy issues, with MS keeping things more abstract around “ethics”: “Microsoft issued a set of principles around uses of AI in the open world. These include fairness, accountability, transparency, and ethics”, and the Google paper placing more emphasis on handling sensitive information, dedicating a test to this topic: “The data pipeline has appropriate privacy controls” (data test 5).

---

## Discussion

A key point of departure between the two papers is that, for the most part, the Google paper treats all tests as equal, or perhaps more accurately, deliberately shies away from attempting to prioritize them. On the other hand, the MS paper highlights differences in perceived challenges by teams with different levels of experience:

*Figure 2*: Results table from Amershi *et al*. (2019)

TABLE II
THE TOP-RANKED CHALLENGES AND PERSONAL EXPERIENCE WITH AI. RESPONDENTS WERE GROUPED INTO THREE BUCKETS (LOW, MEDIUM, HIGH) BASED ON THE 33RD AND 67TH PERCENTILE OF THE NUMBER OF YEARS OF AI EXPERIENCE THEY PERSONALLY HAD (N=308). THE COLUMN *Frequency* SHOWS THE INCREASE/DECREASE OF THE FREQUENCY IN THE MEDIUM AND HIGH BUCKETS COMPARED TO THE LOW BUCKETS. THE COLUMN *Rank* SHOWS THE RANKING OF THE CHALLENGES WITHIN EACH EXPERIENCE BUCKET, WITH 1 BEING THE MOST FREQUENT CHALLENGE.

| | Frequency | | | Rank | | |
|---|---|---|---|---|---|---|
| Challenge | Medium vs. Low | High vs. Low | Trend | Experience Low | Medium | High |
| Data Availability, Collection, Cleaning, and Management | -2% | 60% | | 1 | 1 | 1 |
| Education and Training | -69% | -78% | | 1 | 5 | 9 |
| Hardware Resources | -32% | 13% | | 3 | 8 | 6 |
| End-to-end pipeline support | 65% | 41% | | 4 | 2 | 4 |
| Collaboration and working culture | 19% | 69% | | 5 | 6 | 6 |
| Specification | 2% | 50% | | 5 | 8 | 8 |
| Integrating AI into larger systems | -49% | -62% | | 5 | 16 | 13 |
| Education: Guidance and Mentoring | -83% | -81% | | 5 | 21 | 18 |
| AI Tools | 144% | 193% | | 9 | 3 | 2 |
| Scale | 154% | 210% | | 10 | 4 | 3 |
| Model Evolution, Evaluation, and Deployment | 137% | 276% | | 15 | 6 | 4 |

> *"With teams across the company having differing amounts of work experience in AI, we observed that many issues reported by newer teams dramatically drop in importance as the teams mature, while some remain as essential to the practice of large-scale AI."*
>
> - Amershi *et al*. (2019)

Whilst this isn't exactly prioritization, it's is useful for understanding which challenges are likely to affect which teams, and also there is the implied point that those difficulties highlighted by experienced practitioners are more likely to be show-stoppers:

> *"Challenges around tooling, scale, and model evolution, evaluation, and deployment are more important for engineers with a lot of experience with AI. This is very likely because these more experienced individuals are tasked with the more essentially difficult engineering tasks on their team; those with low experience are probably tasked to easier problems until they build up their experience."*

For all that nuance, it's also important to note that "**Data Availability, Collection, Cleaning, and Management** [was ranked] as the top challenge by many respondents, no matter their experience level" [empasis added]. This corresponds with what many data scientists across industries have found.
Although the Google paper largely avoids value judgments, I think the following points are particularly interesting and may stray into that territory:

- Integration testing (Infra 3) stood out as a test with much lower adoption than most

First, canarying can indeed catch many issues like unservable models, numeric instability, and so forth. However, it typically occurs long after the engineering decisions that led to the issue, so it would be significantly preferable to catch issues earlier in unit or integration tests.

Perhaps the most important and least implemented test is the one for training/serving skew (Monitor 3). This sort of error is responsible for production issues across a wide swath of teams, and yet it is one of the least frequently implemented tests. In part this is because it is difficult, but again, building this into a framework like TFX allows many teams to benefit from a single investment.

Despite its lack of prioritization, to its credit the Google paper has a clear call to action, specifically applying its tests as a checklist. This is something I heartily agree with, as will anyone who is familiar with Atul Gawande's *The Checklist Manifesto*.

It's also useful to observe the links and dependencies across different best practices:

- Without model specification review and version control (model test 1), it would be hard for reproducible training (infra test 1)
- Without reproducible training (infra test 1), the effectiveness and predictability of canary releases (infra test 6) are significantly reduced.
- Without knowing the impact of model staleness (model test 4), it's hard to implement training/serving skew tests (monitor test 2)

The web of interdependencies that I am just scratching the surface of here certainly supports Google's decision to base their rubric scoring approach on the lowest score across all five of their testing categories. To me a key useful heuristic remains reproducibility, as it is a topic that spans the following areas:

- Data Availability, Collection, Cleaning, and Management
- AI Tools
- End-to-end pipeline support
- Model Evolution, Evaluation, and Deployment

These are four of the five top-ranked challenges by highly experienced ML practitioners in the Microsoft paper. The more I study this area, the more I believe that reproducibility is a powerful "north star" which leads to following many of the best practices discussed. I also think it will become increasingly important as regulatory concerns grow.

---

## Conclusion

If you're involved in machine learning systems, read these papers (and similar papers dealing with best practices), so that you can learn from others' mistakes. It really is much more pleasant than making the mistakes yourself. Get a checklist, figure out where the gaps are, and build on the shoulders of the community.