W&B

THE WEIGHTS & BIASES GUIDE TO

Building a tech stack for ML expertise

W&B

Executive Summary

Machine learning is increasingly important for driving business value, but it brings its own set of operational issues. Google, one of the earliest adopters of machine learning, summarized this in their paper “Machine Learning: The High Interest Credit Card of Technical Debt”, where they described the dangers of blindly applying standard engineering practices to machine learning in production.

ML Ops is a new business requirement that has emerged to support enterprise-grade machine learning. These tools provide the groundwork that practitioners need to achieve high efficiency and build reliable models.

In a quickly evolving space, traditional analysts struggle to compile sensible recommendations. For example, Gartner described SAS and MathWorks as the leading data science and machine learning platforms.

We've worked with hundreds of enterprises and observed the differences that great tooling can make in scaling production ML. Not every tool is right for every business, but we present a decision making framework so you can choose the tools that fit your objectives.

W&B

Why your ML Ops tech stack is strategic

Your machine learning models can be a huge competitive advantage or disadvantage. At this point, nearly every successful enterprise has either built software competency or been disrupted. We see the same trend emerging with machine learning, only this time it's unfolding faster.

Some companies choose to outsource all of their machine learning competence to consultants or tools that claim to automate all of machine learning development. We think this is a huge mistake, and we think these companies will gradually disappear. We're focusing on executives that know that they need to build an ML team in-house and support them with high quality infrastructure.

The best companies know that the models and data are the source of their competitive edge, but they only exist because of the ML team and the ML Ops infrastructure.

W&B

Common anti-patterns

We recently talked to a CTO of a large online retailer, and he described how 3 separate teams had all built separate internal tools for experiment tracking and model deployment.

BETTING ON THE WRONG SOFTWARE

We talked to the head of ML of a large tech-forward company that had adopted machine learning early. They built all their tooling around the Caffe framework, which at the time made sense from a purely technical perspective. When the main Caffe author left the project, the framework became increasingly less supported. Now the company wants to transition to PyTorch, but it's hard to resource such a huge undertaking.

INVESTING IN THE WRONG AREAS

We talked to a non-technical executive at a consumer packaged goods company. His ML team told him that building low-level internal ML tooling was a path to long-term strategic advantage and a way to create valuable IP. Unfortunately, after two years and millions of dollars of investment, open source tools such as TensorFlow Extended offer better solutions to the same problems, and he's planning to abandon the internal work.

W&B

Success stories

DEVELOPER-LEAD TOOLING

John Deere heavily invested in flexible internal tooling. For example, they supported TensorFlow and PyTorch at the same time. They let the ML practitioners do lots of experimentation with third party tools and find the best fit. The result is that they've built a strong developer brand, and have been able to attract top-tier talent over their competitors.

DEVELOPMENT VELOCITY

Toyota needed to keep pace with competitors' autonomous vehicle projects. They chose to partner with third-party vendors to accelerate the speed of development.

FOCUS ON DIFFERENTIATION

Insitro needed to connect their machine learning models to real-world wet lab experimentation in a highly regulated industry. Their internal tools teams are focusing attention on the truly custom work needed to piece their advanced pipeline together, and incorporating third-party tools to solve the common machine learning problems.

W&B

Decision points

Until recently, third-party tools were not available to support ML teams, so most tooling was in-house. In-house tools are more flexible and can be viewed as a strategic advantage, but might not be aligned with business objectives.

PLATFORM VS. TOOLKIT

Some companies buy in to complete platforms like DataRobot. They benefit from fast ramp-up with a less experienced technical team. As organizations scale and requirements change, it can be hard to adapt these less flexible systems to new technologies.

For example, Transformers first appeared in 2019 and are now the industry standard for anyone doing natural language processing, yet they're still not supported by popular platforms. Another downside of platform lock-in is the difficulty in hiring. Engineers are attracted to roles that build transferable skills.

IN-HOUSE VS. THIRD PARTY

Many engineers view building internal ML tools as a path towards career advancement and an entry into the lucrative ML space. This is not necessarily a bad thing, but it may not be aligned with your business objectives.

Companies often view building in-house tools as a way to build a strategic advantage around machine learning competence, but actually most tech-forward companies such as Google, Facebook, CapitalOne, and Genentech heavily rely on third party tools and view the data and models as their strategic advantage. As third party tools mature, we expect this trend to expand across industries.

W&B

ML Ops decision framework

Vendor lock-in

Hiring elite talent

Total cost of ownership

Software viability

Visibility

Governance and privacy

W&B

ML Ops decision framework

VENDOR LOCK-IN

Google offers infrastructure in Google Cloud, backs the popular framework TensorFlow, and sells higher-level tooling such as Auto ML. Amazon also sells bare infrastructure in AWS, offers SageMaker for higher level model training abstraction, and sells specialized ML APIs such as Rekognition.

Amazon and Google claim that their software is interoperable, but also have explicitly stated that they use their ML tools as a differentiating wedge to move customers to their infrastructure.

On one hand, Google will naturally offer phenomenal support for TensorFlow models, but recently PyTorch has become increasingly popular, and currently Google support is lacking. Infrastructure and framework-agnostic vendors that only offer one level of the stack represent safer long-term choices.

HIRING ELITE TALENT

It's long been known that the best engineers care about their employer's tech stack. Companies have come to realize that their technical choices have an impact on who they are able to hire. Competent ML practitioners are sought after and hard to find, so this is especially important. In-house tools represent non-transferrable skills, which are particularly unappealing.

Obscure tools such as Jax or MXNet may be sensible technical choices, but put you at the mercy of a small number of specialists. Proprietary tools, especially black-boxes such as Domino or SigOpt, can make developers feel like they're not learning generalizable skills.

W&B

ML Ops decision framework

TOTAL COST OF OWNERSHIP

The real cost of your ML projects is the people, hardware, and data. The cost of tools or platforms will always be small in comparison, and more expensive tools aren't necessarily better than free or open source tools, though they often are.

If your primary cost is people, then you should view tools in the lens of what efficiency improvements are possible. Many ML practitioners say they spend most of their time cleaning and examining data, so an investment in data cleaning and visualization tools could be smart.

If your primary cost is hardware, you should look for tools that make sure your hardware is used efficiently. We've found that when practitioners have easy access to system information, they're empowered to make their processes more efficient, and there is an order of magnitude of possible efficiency gains. For example, most model training runs don't fully use the expensive GPU that they're running on. Most experiments aren't actually necessary with a smarter hyperparameter search scheme. Re-running experiments is not necessary when you have a reliable system of record.

If your primary cost is data, and you've done everything to lower your cost of data, then you should focus on empowering your team with tools that make sure they make full use of the data. Resist the temptation to over-optimize hardware spend.

W&B

ML Ops decision framework

SOFTWARE VIABILITY

There's rapid change in the ML space, and picking a tool from a company that goes out of business or an open source project that stops being maintained is a disaster that you should avoid.

Open source tools aren't a hedge against a company going out of business. Caffe, MXNet, Theano are all examples of once-popular open source tools that have since fallen out of maintenance and left users with a migration nightmare and possible security vulnerabilities.

It might seem safer to go with a big company like Microsoft or IBM, but they have been increasingly willing to deprecate or shut down tools such as MLStudio.

Apart from avoiding lock-in, it's important to look at signals that the technology you're choosing has momentum and an active ecosystem.

W&B

ML Ops decision framework

VISIBILITY

We think that user experience is a highly under-rated aspect of developer tools in general, and ML tools in particular. Many of the high profile ML PR disasters have come from developers not following basic best practices. For example, Microsoft came under scrutiny for releasing a facial recognition model that performed much worse on black people.

Clearly, the developers neglected to look at the class distributions of their input data, a mistake that could have easily been avoided and probably would have been avoided if they had tools that made it easy to visualize their input data. Mistakes like this aren't always so high profile, but always lead to bad outcomes. This is why, even for expert developers, tools that make it effortless to follow best practices are essential for reliable model development and ML safety.

GOVERNANCE AND PRIVACY

Data governance and privacy requirements differ across industries, but in the most meaningful machine learning applications such as healthcare or autonomous vehicles, regulatory compliance is a huge issue.

A common issue in ML is ensuring that a specific piece of data wasn't used in a deployed model. As model pipelines grow in complexity, this can become a real challenge. For example, ML leaders at iRobot detailed the evolution of their machine learning toolkit to support model governance. A good ML Ops stack in any application should make it effortless to know the data provenance of any model in production.

W&B

Components of the superior ML stack

Hardware and infrastructure

Frameworks

Dataset versioning

Experiment tracking

Continuous integration

Production monitoring

W&B

Components of the superior ML stack

HARDWARE AND INFRASTRUCTURE

At the time of writing in late 2020, training on NVIDIA hardware is significantly cheaper than using Azure, AWS, or GCP's hosted cloud GPU/TPU offerings. Even though that's the case, most companies we work with choose to run their training in the cloud for convenience, unless there is a regulatory compliance issue.

Azure, AWS, and GCP's pricing is similar, so most companies choose to train on the same cloud that they use for the rest of their development, but only GCP offers TPUs, so for some use cases their offering is superior.

Building your own hardware is a challenge, but companies like Lambda Labs make it easy to set up an in-house cluster.

FRAMEWORKS

Surprisingly, most of our customers do not use a single framework. Each framework has its own set of advantages, and the frameworks borrow quite a lot from each other, but at this moment the clear leaders are Scikit for traditional machine learning and PyTorch and TensorFlow / Keras for deep learning.

One reason to support multiple frameworks is that there is a huge amount of open source work to build on for any application, and the open source projects are written in different frameworks. Companies that make a seemingly sensible decision to standardize on a single framework often end up undoing that decision when a crucial open-source project is built on a different framework.

W&B

Components of the superior ML stack

DATASET VERSIONING

Some companies have a small amount of incredibly valuable data. For example, for some healthcare applications, 100 records would be an impressive cache of information. For other companies, they have terabytes of data that flow through their system hourly, so it's not even realistic to save all the data that they might use.

Therefore, there's no single best data versioning tool or system, but most companies don't actually systematically do dataset versioning right now which is a recipe for disaster.

Without the dataset that a model was trained on, it's impossible to have reproducibility and it's hard to have any sense of explainability.

EXPERIMENT TRACKING

The essential task of an ML practitioner is to run experiments in the same way that the essential task of a software engineer is to write code. no company would dream of saving their code in an ad-hoc system, yet until recently there wasn't a systematic way to save ML experiments.

A good experiment tracking tool has the opportunity to do more than just save experiments. It can make practitioners exponentially more efficient by helping them sift through the enormous amount of data that's generated as a model trains.

W&B

Components of the superior ML stack

CONTINUOUS INTEGRATION AND DEPLOYMENT

When we talk to ML practitioners, one of the biggest drags on productivity is a small inadvertent change that degrades model performance for the whole team and goes unnoticed for weeks.

That change can invalidate all the downstream results, and this problem isn't unique to ML. It was solved a long time ago with continuous integration systems for software engineering, but model building is fundamentally different than software engineering in that there is not always a clear set of unit tests that need 100% pass rate.

EXPERIMENT TRACKING

The best companies set up a system for automatically testing the latest changes to their codebase, and this needs to happen automatically. Excellent companies retrain their models all the time and fearlessly deploy those new models into production. Real world data distributions change over time, so only deploying once every 6 months due to concerns around the reliability of the model can severely degrade production performance.

Bad tooling will make it scary to deploy a new model to production, so deployments will happen less frequently. In traditional software development, infrequent deployments lead to stagnation and low moral, but in ML the model performance will often degrade. For example, a model trained on tweets from 2019 would perform significantly worse on data from just one year later.

W&B

Components of the superior ML stack

Machine learning is powerful but can be brittle, especially when a model is exposed to data it wasn't trained on. There's two common modes of failure to look out for: unexpected inputs and data drift.

UNEXPECTED INPUTS

New data that is unlike any data the model has seen before can be catastrophic. For example, if you look at the well-known crashes of Teslas in auto-pilot, there's almost always a very unexpected situation like a semi-truck parked perpendicularly across the highway.

The problem is that models don't typically throw errors, so unless you're explicitly monitoring for unexpected inputs, you may not know that there's an issue.

DATA DRIFT

If you deploy a fraud model, attackers can probe it for vulnerabilities and find ways to exploit it. Your model quickly starts performing a lot worse on the real world production data than the data the model was trained on. This failure mode is subtle but also common, where your underlying world changes. For example, seasonality in demand forecasting has long been a known issue. Surprise events like COVID-19 can completely undermine the performance of predictive models.

These issues aren't necessarily hard to spot if you're looking for them, but you need tooling that consistently surfaces changes in input distributions.

W&B

Conclusion

Strong ML Ops are increasingly crucial to support enterprise machine learning. There are dozens of frameworks and tools emerging in this space, and it's challenging to predict which tools will best support your team in the rapidly evolving landscape of ML applications.

Successful leaders making ML tools decisions are taking into account vendor lock-in and avoiding proprietary tools that will fall behind industry standards and make hiring challenging. Though it's tempting to buy into open-source libraries, we've seen enterprise ML teams struggle to migrate projects after these tools fall out of favor.

Prioritizing viable tools that have momentum and traction, a solid ML tech stack will incorporate a reliable system of record that provides visibility across the organization. At the end of the day, these tools should empower ML practitioners to make well-informed decisions and build reproducible models.