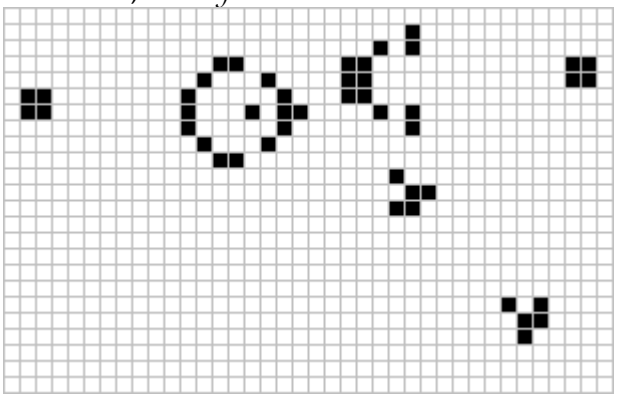
REPLY

Pete Warden's blog

Ever tried. Ever failed. No matter. Try Again. Fail again. Fail better.

The Machine Learning Reproducibility Crisis

March 19, 2018 *By* Pete Warden *in* Uncategorized 42 Comments



Gosper Glider Gun (http://www.conwaylife.com/w/index.php?title=Gosper_glider_gun)

I was recently chatting to a friend whose startup's machine learning models were so disorganized it was causing serious problems as his team tried to build on each other's work and share it with clients. Even the original author sometimes couldn't train the same model and get similar results! He was hoping that I had a solution I could recommend, but I had to admit that I struggle with the same problems in my own work. It's hard to explain to people who haven't worked with machine learning, but we're still back in the dark ages when it comes to tracking changes and rebuilding models from scratch. It's so bad it sometimes feels like stepping back in time to when we coded without source control.

When I started out programming professionally in the mid-90's, the standard for keeping track and collaborating on source code was Microsoft's Visual SourceSafe (https://en.wikipedia.org/wiki/Microsoft_Visual_SourceSafe). To give you a flavor of the experience, it didn't have atomic check-ins, so multiple people couldn't work on the same file, the network copy required nightly scans to avoid mysterious corruption, and even that was no guarantee the database would be intact in the morning. I felt lucky though, one of the places I interviewed at just had a wall of post-it notes, one for each file in the tree, and coders would take them down when they were modifying files, and return them when they were done!

This is all to say, I'm no shrinking violet when it comes to version control. I've toughed my way through some terrible systems, and I can still monkey together a solution using rsync and chicken wire if I have to. Even with all that behind me, I can say with my hand on my heart, that machine

learning is by far the worst environment I've ever found for collaborating and keeping track of changes.

To explain why, here's a typical life cycle of a machine learning model:

- A researcher decides to try a new image classification architecture.
- She copies and pastes some code from a previous project to handle the input of the dataset she's using.
- This dataset lives in one of her folders on the network. It's probably one of the ImageNet downloads, but it isn't clear which one. At some point, someone may have removed some of the images that aren't actually JPEGs, or made other minor modifications, but there's no history of that.
- She tries out a lot of slightly different ideas, fixing bugs and tweaking the algorithms. These changes are happening on her local machine, and she may just do a mass file copy of the source code to her GPU cluster when she wants to kick off a full training run.
- She executes a lot of different training runs, often changing the code on her local machine while jobs are in progress, since they take days or weeks to complete.
- There might be a bug towards the end of the run on a large cluster that means she modifies the code in one file and copies that to all the machines, before resuming the job.
- She may take the partially-trained weights from one run, and use them as the starting point for a new run with different code.
- She keeps around the model weights and evaluation scores for all her runs, and picks which weights to release as the final model once she's out of time to run more experiments. These weights can be from any of the runs, and may have been produced by very different code than what she currently has on her development machine.
- She probably checks in her final code to source control, but in a personal folder.
- She publishes her results, with code and the trained weights.

This is an optimistic scenario with a conscientious researcher, but you can already see how hard it would be for somebody else to come in and reproduce all of these steps and come out with the same result. Every one of these bullet points is an opportunity to inconsistencies to creep in. To make things even more confusing, ML frameworks trade off exact numeric determinism for performance, so if by a miracle somebody did manage to copy the steps exactly, there would still be tiny differences in the end results!

In many real-world cases, the researcher won't have made notes or remember exactly what she did, so even she won't be able to reproduce the model. Even if she can, the frameworks the model code depend on can change over time, sometimes radically, so she'd need to also snapshot the whole system she was using to ensure that things work. I've found ML researchers to be incredibly generous with their time when I've contacted them for help reproducing model results, but it's often months-long task even with assistance from the original author.

Why does this all matter? I've had several friends contact me about their struggles reproducing published models as baselines for their own papers. If they can't get the same accuracy that the original authors did, how can they tell if their new approach is an improvement? It's also clearly concerning to rely on models in production systems if you don't have a way of rebuilding them to cope with changed requirements or platforms. At that point your model moves from being a high-interest credit card of technical debt (https://research.google.com/pubs/pub43146.html) to something more like what a loan-shark offers. It's also stifling for research experimentation; since making changes to code or training data can be hard to roll back it's a lot more risky to try different variations, just like coding without source control raises the cost of experimenting with changes.

It's not all doom and gloom, there are some notable efforts around reproducibility happening in the community. One of my favorites is the TensorFlow Benchmarks (https://www.tensorflow.org/performance/benchmarks) project Toby Boyd's (https://twitter.com/tobyjboyd) leading. He's made it his team's mission not only to lay out exactly how to train some of the leading models from scratch with high training speed on a lot of different platforms, but also ensures that the models train to the expected accuracy. I've seen him sweat blood trying to get models up to that precision, since variations in any of the steps I listed above can affect the results and there's no easy way to debug what the underlying cause is, even with help from the authors. It's also a never-ending job, since changes in TensorFlow, in GPU drivers, or even datasets, can all hurt accuracy in subtle ways. By doing this work, Toby's team helps us spot and fix bugs caused by changes in TensorFlow in the models they cover, and chase down issues caused by external dependencies, but it's hard to scale beyond a comparatively small set of platforms and models.

I also know of other teams who are serious about using models in production who put similar amounts of time and effort into ensuring their training can be reproduced, but the problem is that it's still a very manual process. There's no equivalent to source control or even agreed best-practices about how to archive a training process so that it can be successfully re-run in the future. I don't have a solution in mind either, but to start the discussion here are some principles I think any approach would need to follow to be successful:

- Researchers must be able to easily hack around with new ideas, without paying a large "process tax". If this isn't true, they simply won't use it. Ideally, the system will actually boost their productivity.
- If a researcher ~~gets hit by a bus~~ founds their own startup, somebody else should be able to step in the next day and train all the models they have created so far, and get the same results.
- There should be some way of packaging up just what you need to train one particular model, in a way that can be shared publicly without revealing any history the author doesn't wish to.
- To reproduce results, code, training data, and the overall platform need to be recorded accurately.

I've been seeing some interesting stirrings in the open source and startup world around solutions to these challenges, and personally I can't wait to spend less of my time dealing with all the related issues, but I'm not expecting to see a complete fix in the short term. Whatever we come up with will require a change in the way we all work with models, in the same way that source control meant a big change in all of our personal coding processes. It will be as much about getting consensus on the best practices and educating the community as it will be about the tools we come up with. I can't wait to see what emerges!

42 responses

Pej Hamidi says:
March 19, 2018 at 3:15 pm
Compellon solved this problem with their clear box collaborative AI platform. Big name clients are using it in multiple domains.

superhumanvision says:
March 19, 2018 at 6:14 pm
Good article. I am waiting too

Doppefrog says:
March 19, 2018 at 10:28 pm

Sounds like the problem is, as much as anything, a lack of discipline and rigor when developing and testing. Sounds like the field could learn a lot from the software engineering field.

Chang Hsin Lee (@ChangLeeTW) says:
March 20, 2018 at 12:55 am
Researchers are not rewarded by taking a software build approach — why should people structure their projects so they can do a 1-click build and reproduce everything when the norm in academia is publish or perish? Maybe having a journal that dedicate to reproducible research would help?

ALEXNET says:
March 20, 2018 at 2:51 am
Maybe the issue here is that most of these researchers do not care about the robustness of model/cleanliness of code or training processes as what they are judged on are often metrics that do not depend on aforementioned factors

Leon Derczynski says:
March 20, 2018 at 8:08 am
Right. This is why in data management, VLDB has had a replication track, and at COLING, we have a reproducibility track. Being able to reproduce and replicate results (or not) is a sign of maturity – ask the life sciences – and those who try, should be rewarded.

Pete says:
March 20, 2018 at 9:07 am
Results that cannot be reproduced? Sounds like a failed experiment, Or a misconceived model.

szakib says:
March 20, 2018 at 10:12 am
The model params and the training data are somewhat tricky, but there is no excuse for keeping the code outside of version control.

lewiscowles2015 says:
March 20, 2018 at 10:30 am
I really hope this isn't the case across the industry. I'm anti-ML in the tensor-Flow bayesian black-box sense, but I do have high hopes for a future where the machine will produce step-by-step flows (even of high-level) predictable steps taken to reach conclusions (I see that as the only way we can learn from the learning machines, which is a very nice application).

Would using a distributed FS help with the changing of files? You could grant everyone read-only access to an S3 or S3 compatible object store for example and then once training was complete, nobody could modify the data. Complete first iteration, if it passes, freezes by removing write access to that S3 bucket, then repeat until done {n} times?

Even using something that doesn't trivially support deletion could help and there are lots of solutions for systems where granular permissions around CRUD operations are available. Either that or export a build artefact before running the ML, so that you have a time-stamp archive for every run, that is imported to a VM (slows down, but increases reproducibility) and the environment can be exported, recorded, snapshot, etc.

An interesting yet terrifying article.

itsagoodbrain says:
March 20, 2018 at 10:39 am

A really well written article and I appreciate your use of she for your example. Upon reading I almost got upset because it was so detailed and I was wondering who you'd thrown under the bus for doing what we all do when training. Once I got far enough in to realize it was so detailed because it was the best case scenario, I can genuinely say I was mad at myself instead for jumping to conclusions that you'd be using a she as a bad example. Look forward to future posts.

Ardax says:
March 20, 2018 at 11:20 am
Hi there Pete. I hear you. I suppose we need some kind of a mini operating system that is specifically designed to maintain all versions of data and all its meta data (e.g. info about how it is created). Much like a git. Certainly there is a lot to brainstorm on that idea. Along the same lines, last year I wrote a JS+PHP+MongoDB based tool to keep/manage/explore the experiment results and their parameters. That really helped me run hundreds even thousands of experiments without worrying about how to keep track of the results. It is funny that there is not even such tools around. Not to my knowledge at least. For those who are interested, it is at github.com/ardax/xDB.

> Alice says:
> March 22, 2018 at 11:25 am
> Hi Ardax!
>
> We just launched an OSS project called Dotmesh (www.dotmesh.com) that does exactly that. We're doing a research spike at the moment into the world of Data Science and Machine Learning to see if we can help. We are very keen to speak to anyone who might be happy to tell us about their work practices and challenges around data management. 🙂
>
> Alice

mourad says:
March 20, 2018 at 4:09 pm
You have to check https://github.com/polyaxon/polyaxon, it tries to solve most of the reproducibility issues mentioned in this post.

Siebert Looije says:
March 20, 2018 at 7:13 pm
Hi,
Thanks for your blog. Could the new service 'sagemaker' of AWS not be a solution for these problems ? We are starting to use it and it looks very promising

Pingback: 机器学习研究重现难，难于上青天 2018-03-19 – Androidev

Pingback: Daily Reading #261 | thinkpatriot

Pingback: Links 3/21/18 | Mike the Mad Biologist

Pingback: Distilled News | Data Analytics & R

dvaida says:
March 23, 2018 at 9:30 am
https://dataversioncontrol.com

Nathan Glenn says:
March 26, 2018 at 7:44 am
Fixing the slight changes due to randomization should be a low-hanging fruit; just use a pseudo-random number generator and publish your seed (you could just always use 0).

Pingback: 3 Tips in Training Machine Learning for Security Work - Security Boulevard

Pingback: HRIntelligencer v2.13 | HR Examiner

Pingback: It's time to address the reproducibility crisis in AI – Technology NEWS

Pingback: It's time to handle the reproducibility disaster in AI - All of it

Pingback: It's time to address the reproducibility crisis in AI – Viral Facts

Pingback: The Machine Learning Reproducibility Crisis

Guy says:
May 1, 2018 at 7:04 pm
We have developed a platform for this (and more) and just came out of stealth! Look for allegro.ai (www.allegro.ai) and drop us a note

rohan says:
May 2, 2018 at 6:57 am
Nice article, though I do not have any experience with machine learning and its related fields.

How about having a meta-model, i.e. a model which is trained to do all above things that you have mentioned and people then quering this model to get inputs if they want to reproduce earlier results.

Pingback: Building trust in the decision process. - Ugly Research

Pingback: Data story snafus, plus cognitive bias and fixing the tech diversity problem.

Pingback: La crise de la reproductibilité de l'apprentissage automatique – Datakeo

jane says:
June 1, 2018 at 3:37 pm
Comet.ml is pretty awesome

Pingback: reconstitute the world « Bethany Nowviskie

Abid says:
June 29, 2018 at 9:34 pm
Does reproducibility with respect to throughput and epoch speed also matter? Recently, I ran Tensorflow on 128 GPUs and got different performance when ran at different times.

Pingback: newdarkage: The Machine Learning Reproducibility… | chumbo news

Sandro says:
October 3, 2018 at 12:34 pm
Reproducibility is indeed a key challenge today in data science. It is one of the main reason why the impact of data science is limited, both in the academic world and in the industry. One solution is the RENKU, an open source solution. You can find more information here:
https://datascience.ch/renku-platform/

Pingback: Reproducibility Crisis in Data Science - This Week in Machine Learning & AI

lukas biewald says:
February 6, 2019 at 1:40 am

I've been working on this problem for the last year at Weights and Biases (wandb.com) – I would love anyone's feedback who wants to give it a try.
Pingback: Start Up No.1,005: Google backtracks on Chrome adblock block, make your own cloud, Apple hires to up IoT game, how AI is messing up science, and more | The Overspill: when there's more that I want to say

Pingback: ソフトウェア開発者からデータサイエンティストにキャリアチェンジしたいあなたのためのベストなリソース【前編】| 人工知能ニュースメディア AINOW

Pingback: AI Weekly: AI research still has a reproducibility problem | TechieTricks.com

Pingback: Beyond File Soup: Data Science Reproducibility