GNG: Data-Centric AI Engineering

Arumoy Shome

[2022-08-30 Tue]

This documents presents my research portfolio for the GNG meeting. The document presents an introduction to the research field of Software Engineering for Machine Learning (SE4ML), the intended research field for this thesis. An overview of the state-of-the-art is presented next followed by the research topic of choice. The document concludes with contributions that I have made so far both scientific & supplemental. I also list ideas that were unsuccessful along with ideas we wish to pursue in the future.

1 Background & Motivation

Machine Learning and database systems are well established research fields. Since ML models & datasets go hand-in-hand, their intersection has been well studied. However, with the growing adoption of ML beyond an academic setting, a need for engineering best practices is felt. Although decades of research exists for traditional rule-based software systems, the introduction of Software Engineering for ML systems has been a fairly recent event in the research community.

Figure 1 presents an overview of the Machine Learning Lifecycle. Data analysis is a critical and dominant stage of the machine learning lifecycle. Once the data is collected, most of the work goes into studying and wrangling the data to make it fit for training. A highly experimental phase follows where a model is selected and tuned for optimal performance. The final model is then productionised and monitored constantly to detect data drifts and drop in performance [9, 5, 2, 10].

The *data science* stage is a typical ML workflow which tends to dominate blogposts shared on the internet and also what we commonly see in an academic setting. However as seen in 2, the ML workflow tends to be a very small part of a much larger system.

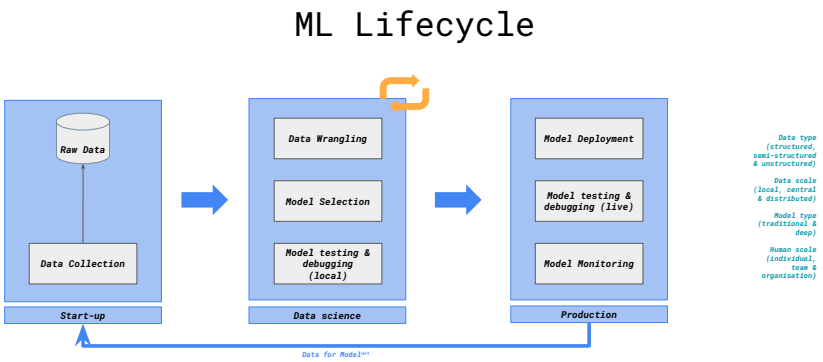


Figure 1: ML Lifecycle

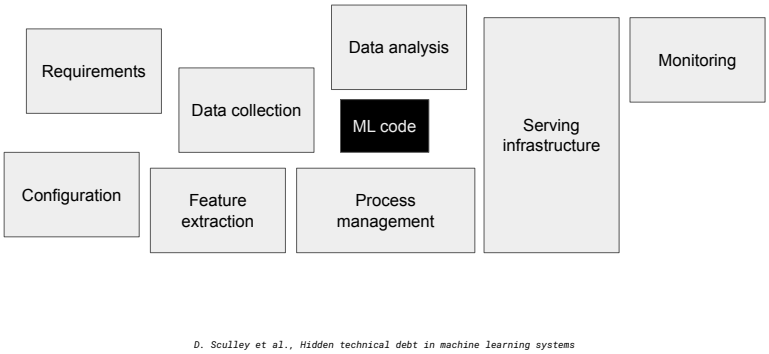


Figure 2: Holistic View of ML Systems

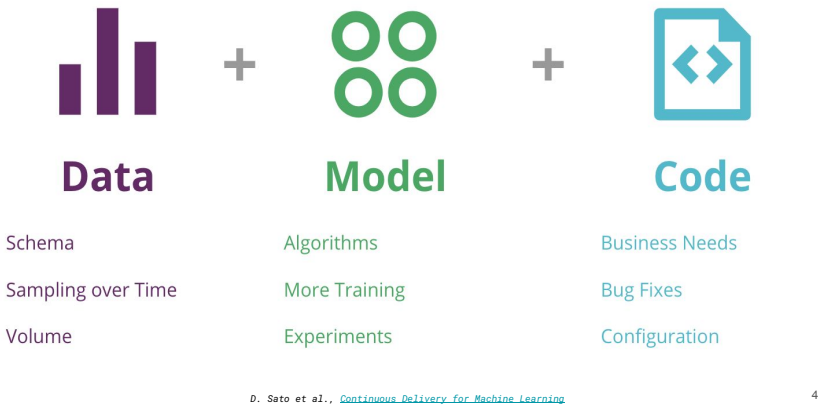


Figure 3: ML Pillars of Change

When compared to traditional software, the feedback loop of a machine learning system is longer. While traditional software primarily experiences change in *code*, a machine learning system matures through changes in *data, model & code* [9]. Given the highly tangled nature of machine learning systems, a change in any of the stages of the lifecycle triggers a ripple effect throughout the entire pipeline [10]. Testing such changes also becomes challenging since all three components need to be tested. Besides the traditional test suites, a full training-testing cycle is required which incurs time, resource and financial costs. The surrounding infrastructure of a machine learning pipeline becomes increasingly complex as we move towards a productionised model. Thus catching potential problems in the early, upstream phase of data analysis becomes extremely valuable as fixes are faster, easier and cheaper to implement.

AI has had a significant impact on the technology sector due to the presence of large quantities of unbiased data [7]. But AI's true potential lies in its application in critical sectors such as healthcare, wildlife preservation, autonomous driving, and criminal justice system [3]. Such high-risk domains almost never have an existing dataset and require practitioners to collect data. Once the data is collected, it is often small and highly biased. While AI research is primarily dominated by model advancements, this new breed of *high-stakes AI* supports the need for a more data-centric approach to AI [6, 8, 11].

ML tends to be very data-centric in nature and majority of the work

involves working with the data. When we put a software engineering lens on ML problems, **data becomes equivalent to code**. While software engineers have several tools & techniques to aid them in their day-to-day lives, the same cannot be said for ML practitioners.

2 Scientific Contributions

This section presents the scientific contributions that we have made so far. I also include the ideas which we are currently working on, ideas that we wish to pursue in the future & ideas which were unsuccessful.

2.1 DONE shome2021data

Since the study of software systems with machine learning components is a fairly young discipline, resources are lacking to aid practitioners in their day-to-day activities. The highly data-driven nature of machine learning makes data equivalent to code in traditional software. The notion of code smells is critical in software engineering to identify early indications of potential bugs, sources of technical debt and weak design choices. Code smells have existed for over 30 years. A large body of scientific work has catalogued the different smells, the context in which they occur and their potential side-effects. To the best of our knowledge, such a catalogue however did not exist for data science.

This study identified the recurrent data quality issues in public datasets. Analogous to code smells, we introduced a novel catalogue of data smells that can be used to indicate early signs of problems or technical debt in machine learning systems. To understand the prevalence of data quality issues in datasets, we analysed 25 public datasets and identified 14 data smells.

The paper was accepted at The 1st International Conference on AI Engineering '22, which was co-located with ICSE '22.

2.2 CANCEL shome2022robust

As seen in Figure 4 & 5, there may be several sources of data for the model. For instance, there may be a web service which is continually scrapping the internet for data, or we may have data stored in a database, a data warehouse or data lake. In addition, we may have several other systems with or without ML components which our system communicates with. For instance, our system may rely on data from another service. In return, other services may depend on the predictions from our system.

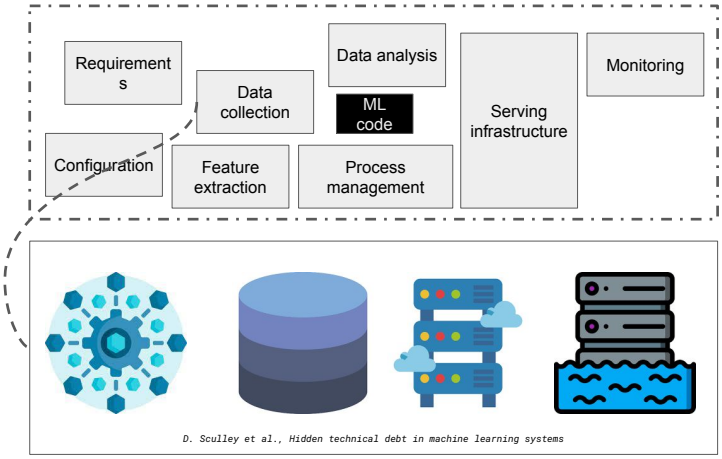


Figure 4: Tangled Systems in ML 1

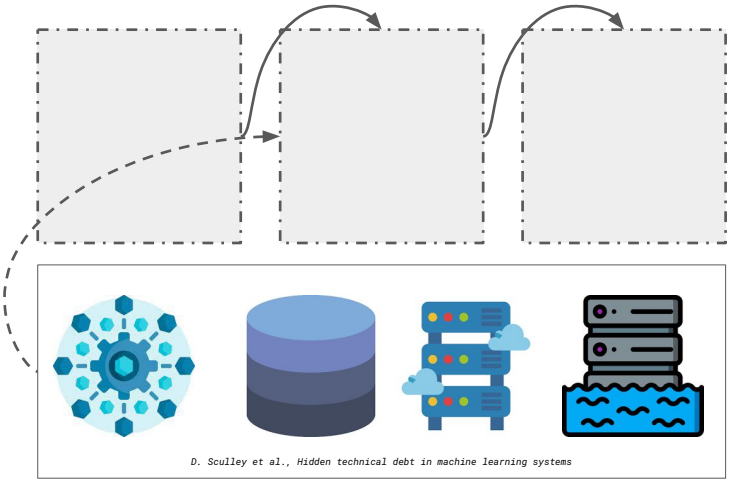


Figure 5: Tangled Systems in ML 2

Thus ML pipelines are inherently complex and tangled. They consist of several stages and the ML model work tends to be a small part of a much larger system. A small change or bug in any of the stages ripples throughout the entire pipeline. Therefore, we cannot make implicit assumptions regarding the quality of the data.

Although data validation libraries exist, currently there are no tools to aid practitioners in writing robust data validation rules. The motivation here is that data will constantly evolve and change as a reflection of the real world. However, this requires a lot of upkeep of data validation rules which are brittle and keep triggering false alerts. Put another way, we want to reduce false positive alerts to on-call engineers to reduce waste of human time & effort.

We started by analysing the data validation rules produced by the Tensorflow Data Validation (TFDV) tool on 218 public datasets. TFDV flagged 74 datasets with anomalies out of 218. We manually analysed the top 5 datasets with the most number of anomalies. However, the results were not fruitful from a research perspective. The premise of TFDV is that data-centric work requires human-in-the-loop approach. The tool provides a basic set of data validation rules, but the practitioners are still expected to write custom validation rules based on domain knowledge.

Next, we tried searching for existing data validations rules from public ML projects. This yet again was unfruitful since there is a lack of public projects which utilise this tool. Projects that do use data validation perhaps cannot make their validation rules public due to privacy & legal obligations.

Our final, albeit unsuccessful attempt was to work with external collaborators in possession of a ML pipeline running in production. We reached out to several researchers & the Dutch Ministry of Transportation & Environment.

2.3 TODO shome2022qualitative

Prior work have conducted analysis of data transformations & their impact on the fairness of the ML pipeline [1]. Work also exists on analysing impact of bias mitigation strategies at different stages of the pipeline and their impact on the overall fairness of the pipeline [4].

However, no work has been done to identify how much of the fairness issues identified using both the model & the data, can be identified only using the data. In other words, when can we adopt a data-centric approach to testing for fairness? And when do we require a model as well to make sure our predictions are not biased?

Although our methodology will be similar to the existing work (primarily that of [1]), our research question is entirely different. We are hoping to contribute a systematic methodology for testing for fairness in ML pipelines. With our work, practitioners can choose the best testing strategy given a fairness issue.

2.4 LATER shome202xtbd

A natural extension to shome2022qualitative is to conduct a qualitative analysis of bias mitigation techniques. The objective here would be to develop a systematic approach to fixing fairness issues in ML pipelines. Given a fairness issue, we want to aid practitioners in determining the best strategy for fixing the issue.

3 Supplemental Contributions

This section presents some contributions I have made outside of my immediate field of research.

- I was a student volunteer at ICSE '22
- I was a teaching assistant for the REMLA course this year. Besides supervising a team of students I also gave a guest lecture on Data Validation for ML.
- I have open sourced the data smells catalogue. I am (slowly) working towards setting up a Github repository to accept contributions.
- I have given a few talks to the group on other topics of interest.
- Caro & I organised SERG's very first Bib meetup where we discussed tools & techniques to manage information, notes & knowledge. Our very own Diomidis was the guest speaker at this event.

4 Appendix

4.1 DE course overview

Figure 6 presents an overview of the DE courses completed so far. I have completed half of the required credits from the research competency and transferable skill categories. Although I am lagging behind in the discipline

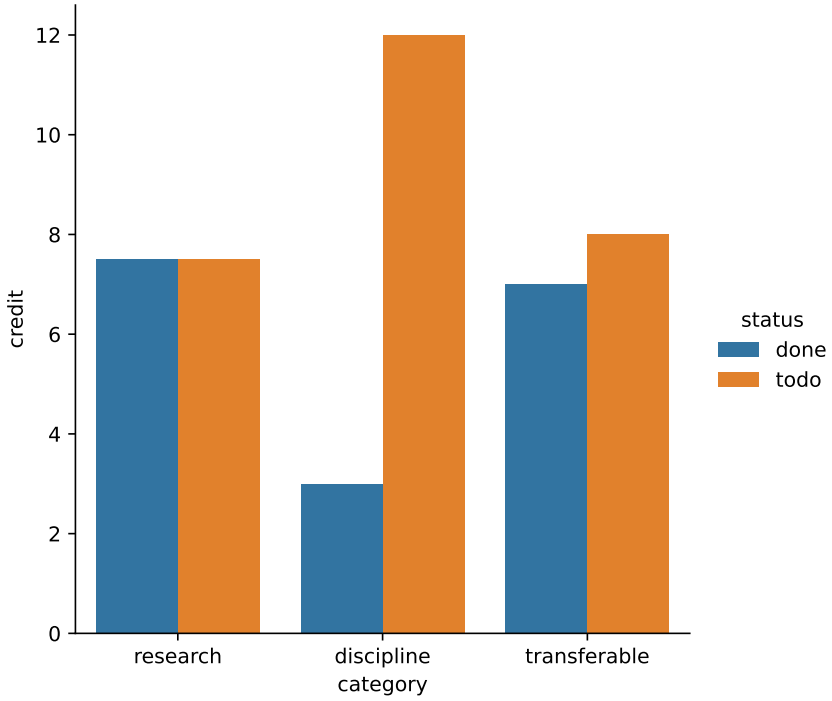
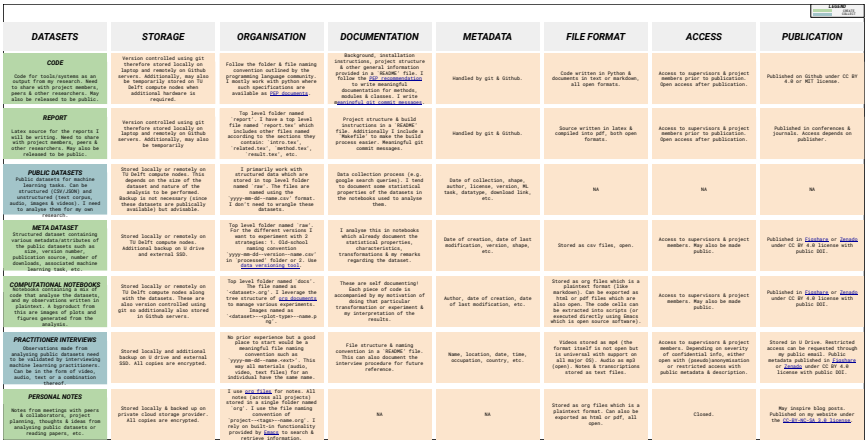


Figure 6: DE Course Overview

skills category, I intend to catch up by attending summer/winter schools in the second half of my Phd.

4.2 Data Management Plan



DATASETS	STORAGE	ORGANISATION	DOCUMENTATION	METADATA	FILE FORMAT	ACCESS	PUBLICATION
CODE Code for tools/systems as an outcome of research. Needs to be shared with project members, peers & other researchers. May also be released to be public.	Version controlled using git therefore stored locally on laptop and remotely on Github servers. Additionally, may also be temporarily stored on TU Delft compute nodes as additional hardware is required.	Follow the folder & file naming convention outlined by the programming language community. I mostly work in python where such specifics are available as PEP documents.	Background, installation instructions, project structure & build instructions are provided in a 'README' file. I follow the PEP recommendation to write meaningful documentation for methods, modules & classes. I write meaningful git commit messages.	Handled by git & Github.	Code written in Python & documents in text or markdown, all open formats.	Access to supervisors & project members prior to publication. Open access after publication.	Published on Github under CC BY 4.0 or MIT license.
REPORT Latex source for the reports I write. Needs to be shared with with project members, peers & other researchers. May also be released to be public.	Version controlled using git therefore stored locally on laptop and remotely on Github servers. Additionally, may also be temporarily	Top level folder named 'report'. I have a top level file named 'report.tex' which includes other files named according to the chapters they contain: 'intro.tex', 'related.tex', 'method.tex', 'result.tex', etc.	Project structure & build instructions in a 'README' file. Additionally I include a Makefile to help build & process easier. Meaningful git commit messages.	Handled by git & Github.	Source written in latex & compiled into pdf, both open formats.	Access to supervisors & project members prior to publication. Open access after publication.	Published in conferences & journals. Access depends on publisher.
PUBLIC DATASETS Public datasets for machine learning tasks. Can be structured (CSV/JSON) and unstructured (text corpus, audio, images & videos). I need to analyse them for my own research	Stored locally or remotely on TU Delft compute nodes. This depends on the size of the dataset and nature of the project I am working on. Backup is not necessary (since these datasets are publically available) but advisable.	I primarily work with structured data which are stored in top level folder named 'raw'. The files are sometimes named in 'yyyy-mm-dd--name.csv' format. I don't need to wrangle these datasets.	Data collection process (e.g. google search queries). I tend to document some statistical properties, issues & my remarks in the notebook used to analyse them.	Date of collection, shape, author, license, version, ML task, datatype, download link, etc.	NA	NA	NA
META DATASET Structured dataset containing various metadata/attributes of the public datasets such as size, version number, publication source, number of downloads, associated machine learning task, etc.	Stored locally or remotely on TU Delft compute nodes. Additional backup on Google drive and external SSD.	Top level folder named 'raw'. For the different versions I want to experiment with I strategies: 1. Old-school naming convention such as 'yyyy-mm-dd--version--name.csv' in 'processed' folder or 2. Use data versioning tool.	I analyse this in notebooks which already document the statistical properties, issues, cleaning & transformations & my remarks regarding the dataset.	Date of creation, date of last modification, version, shape, etc.	Stored as csv files, open.	Access to supervisors & project members. May also be made public.	Published in Figshare or Zenodo under CC BY 4.0 license with public DOI.
COMPUTATIONAL NOTEBOOKS Jupyter notebooks with code that analyse the datasets, and my observations written in plaintext. A byproduct from this are images of plots and figures from the analysis.	Stored locally or remotely on TU Delft compute nodes along with the datasets. These are also version controlled using git to additionally also stored in Github servers.	Top level folder named 'docs'. The file named as '<dataset>.org'. I leverage the tree structure of org documents to manage my experiments. Images named as '<dataset>--<plot-type>--<name.png'.	These are self documenting! Each piece of code is accompanied by my motivation of doing that particular transformation or experiment & my interpretation of the results.	Author, date of creation, date of last modification, etc.	Stored as org files which is a plaintext format (like markdown). Can be exported as html or pdf files which are open. Code in org files can be extracted into scripts (or executed directly using Emacs which is open source software).	Access to supervisors & project members. May also be made public.	Published in Figshare or Zenodo under CC BY 4.0 license with public DOI.
PRACTITIONER INTERVIEWS Observations made from analysing public datasets need to be verified by practitioners working machine learning practitioners. Can be in the form of video, audio, text or a combination thereof.	Stored locally and additional backup on U drive and external SSD. All copies are encrypted.	No prior experience but a good place to start would be a meaningful file naming convention such as 'yyyy-mm-dd--name.org'. This may all materials (audio, video, text files) for an individual have the same name.	File structure & naming convention in a 'README' file. This can also document the interview procedure for future reference.	Name, location, date, time, occupation, country, etc.	Videos stored as mp4 (the format itself is not open but is universal with support on all OS), Audio stored as ogg (open), Notes & transcripts stored as text files.	Access to supervisors & project members. Depending on severity of confidential info, either open access after anonymisation or restricted access with public metadata & description.	Stored in U Drive. Restricted access can be requested through my public email. Published in conferences & journals. Figshare or Zenodo under CC BY 4.0 license with public DOI.
PERSONAL NOTES Notes from meetings with peers & collaborators, project planning, thoughts & ideas from analysing public datasets or reading papers, etc.	Stored locally & backed up on private cloud storage provider. All copies are encrypted.	I use org files for notes. All notes (across all projects) are stored in 'notes/notes.org'. I use the file naming convention of 'project--<tags>--name.org'. I rely on built-in functionality provided by Emacs to search & retrieve information.	NA	NA	Stored as org files which is a plaintext format. Can also be exported as html or pdf, all open.	Closed.	May inspire blog posts. Published on my website under the CC BY-NC-SA 3.0 license

Figure 7: Data Flow Map

Figure 7 presents the data flow map which provides an overview of how various data generated during the course of this Phd will be handled.

References

[1] Biswas, S., Wardat, M., and Rajan, H. The art and practice of data science pipelines: A comprehensive study of data science pipelines in theory, in-the-small, and in-the-large. In *2022 IEEE/ACM 44th International Conference on Software Engineering (ICSE)* (2022), IEEE, pp. 2091–2103.

[2] Bosch, J., Olsson, H. H., and Crnkovic, I. Engineering ai systems: A research agenda. In *Artificial Intelligence Paradigms for Smart Cyber-Physical Systems*. IGI Global, 2021, pp. 1–19.

[3] Crawford, K. *The Atlas of AI*. Yale University Press, 2021.

[4] Ghai, B., Mishra, M., and Mueller, K. Cascaded debiasing: Studying the cumulative effect of multiple fairness-enhancing interventions. *arXiv preprint arXiv:2202.03734* (2022).

[5] Hutchinson, B., Smart, A., Hanna, A., Denton, E., Greer, C., Kjartansson, O., Barnes, P., and Mitchell, M. Towards accountability for machine learning datasets: Practices from software engineering and infrastructure. In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency* (2021), pp. 560–575.

[6] Kshirsagar, M., Robinson, C., Yang, S., Gholami, S., Klyuzhin, I., Mukherjee, S., Nasir, M., Ortiz, A., Oviedo, F., Tanner, D., et al. Becoming good at ai for good. *arXiv preprint arXiv:2104.11757* (2021).

[7] Ng, A. A chat with andrew on mlops: From model-centric to data-centric ai, 2021. Accessed on [2022-01-17 Mon].

[8] Sambasivan, N., Kapania, S., Highfill, H., Akrong, D., Paritosh, P., and Aroyo, L. M. "everyone wants to do the model work, not the data work": Data cascades in high-stakes ai. In *proceedings of the 2021 CHI Conference on Human Factors in Computing Systems* (2021), pp. 1–15.

[9] Sato, D., Wilder, A., and Windheuser, C. Continuous delivery for machine learning.

[10] Sculley, D., Holt, G., Golovin, D., Davydov, E., Phillips, T., Ebner, D., Chaudhary, V., Young, M., Crespo, J.-F., and Dennison, D. Hidden technical debt in machine learning systems. *Advances in neural information processing systems 28* (2015), 2503–2511.

[11] Zhang, J. M., Harman, M., Ma, L., and Liu, Y. Machine learning testing: Survey, landscapes and horizons. *IEEE Transactions on Software Engineering* (2020).