Assignment, WASP Software Engineering Course Module 2024

Name: Newton Mwai
University: Chalmers University of Technology
Division: Data Science and AI
Email: mwai@chalmers.se

1. Research Background: Machine learning for improved sequential decision-making in healthcare with historical data

The primary objective of my research is to leverage historical data in sequential decision making in healthcare. My specific methods are in design of multi-armed bandit (MABs) algorithms which are suitable for consideration in treatment personalization of chronic diseases such as Alzheimer's disease (AD) and rheumatoid arthritis (RA). In these diseases, since the disease cannot be eliminated, treatment involves easing the symptoms like improving cognitive ability in AD or easing the pain in RA. For this reason, treatments considered are primarily treatments with short-term effects and this allows for analysing treatment as a sequential decision making process with MABs. The key challenge with many sequential decision making algorithms in reinforcement learning is sample innefficiency (MABs are a special case of reinforcement learning where the state doesn't evolve with the actions). However, it is often the case that many hospitals and electronic health registries have collected large amounts of data from histories of patioent treatment. My research considers how such historical data can be used to improve sample efficiency. One way to personalise treatments with historical data would be to train agents on historical data and deploy them straight to clinical settings with new unseen patients, but the challenge of distribution shift is quite crippling. A more feasible approach is to think about what information could be extracted from the historical data that could be useful for new, unseen patients. One example[1] that we leverage is patient similarity in the form of a latent variable model (LVM). This entails using the knowledge that patients cluster somehow in disease state and treatment response. In AD for example, there are studies showing that AD disease can be grouped into a number of distinct states, where patients with the same underlying state respond similarly to treatments. If such an LVM can be estimated, when a new patient is to be treated, the task becomes one of estmating which state a patient belongs to, and assigning the optimal treatment which is known from the LVM. Another direction[2] of my research is designing multi-armed bandit algorithms for treatment personalisation that are realistic in the real world. For instance, when treating patients, there could be an incentive to assign the same treatment successively before the treatment takes effect, but also find the optimal treatment optimally in this setting.

2. Reflection on principles/ideas/concepts/techniques from Robert's lectures

One of the key concepts that I am now adeptly aware of and that I'm working to improve on in my research workflow is the idea of "Hidden Technical Debt" in machine learning systems. As my research is primarily in the design of algorithms, I find recurring evidence of glue code in algorithms implemented with very specific ptimization libraries, dead experimental codepaths that are challenging to discern especially when going back to a code base, and pipeline jungles especially when collaborating with other researchers who use other heterogeneeous research processes. Data dependency is also a major challenge. Given that I experimentally validate my research with AD data, the results are tightly coupled to this particular disease and I often think about the translational value in other diseases.

Another idea that was interesting was that of using "surprise adequacy" in designing test cases for machine learning systems. This idea has parallels to some theoretical ideas in MAB design where the theoretical guarantees have typically considered the "worst case" that is observered very rarely in practice and so the theoretical guarantees provided are vacuous. There is a shift to using "instance-specific" guarantees which consider the actual characteristics of a problem setting, yielding more realistic guarantees. Designing testing using surprise adequacy seems to encourage testing with data that is actually encountered in the real world with high probability, therefore more meaningful and robust testing specifications.

3. Reflection on principles/ideas/concepts/techniques from guest lectures

Per Lenberg's guest lecture was quite an interesting introduction to behavioural software engineering. It provided insights into thinking about designing software systems in an organisation while considering the human aspect. It was a valuable reminder that people in an organisation do not always behave ideally, but rather they have unobservable behavioural motivations guided by their inter-personal relationships and their own interests. The subsequent lecture in behavioural software engineering was valuable in situating the context of working with humans and understanding software engineering from a psychological perspective, from the organisation, group and individual level.

The volvo guest lecture was also interesting to think about building software that is designed for complex systems like vehicles, and thinking about the heterogeneity of the specifications and data from many different domain engineers who communicate differently. It was interesting to hear about their process of unifying heterogeneously collected data that relates to the same system using LLMs, and seeing how machine learning can add value to software systems by targeting very specific bottlenecks in a bigger complex engineering system.

4. Discussion of papers published in CAIN conferences

i) Can causality accelerate experimentation in software systems?

Paleyes, Andrei, Han-Bo Li, and Neil D. Lawrence

The main idea of this proposal paper is using causal inference techniques with dataflow architectures to enable less costly experimentation of complex software systems. By using dataflow graph of a software system as a causal graph and code changes as interventions, the authors propose that software engineeers can estimate the downstream effects of code changes without the need for extensive and costly real-world experimentation. This approach could significantly reduce the time and resources required for testing. The main benefits of this idea in AI engineering is speeding up testing of code updates, while minimizing the risks associated with unintended side effects of changes in software components.

The authors' idea of using causal inference in dataflow architectures resonates with my research on utilizing historical data for sequential decision-making in healthcare. In both cases, the challenge is to make informed decisions based on the available data, while managing the uncertainty and potential risks associated with those decisions. Just as the paper proposes using causal models to anticipate the effects of code changes, my research aims to use LVMs which often are motivated and constructed with underlying causal graphs. Both approaches seek to enhance the efficiency and reliability of decision-making processes with causality inspired themes.

In a larger AI-intensive software project, the ideas from this paper could be integrated to improve the robustness and efficiency of the system for instance by anticipating the effects of changes to particular components, like data acquisition modules, model construction models etc. For instance, it would be faster and cheaper to estimate the effects of updating software that collects data more accurately to estimate the effect of personalisation algorithms, even before doing the updates if the whole system is considered as a causal graph. This would allow for cheaper and faster deployment of updates, while also ensuring that any potential negative impacts are identified and mitigated early.

My research is already heavily inspired by causal inference ideas, and thinking of the whole system as a causal graph is apppealing to me. Ideas in intervention and experimentation could potentially be applied to the whole system, and even further, sequential code updates could be modelled in a reinforcement learning setting, even potentially using MABs.

ii) Automatically Resolving Data Source Dependency Hell in Large Scale Data Science Projects.

Boué, Laurent, Pratap Kunireddy, and Pavle Subotić.

The core idea of this paper is the development of an automated framework that maps and monitors data source dependencies within ML models by leveraging activity graphs. In large-scale ML operations, the complexity and volume of data sources can lead to “data source dependency hell” which occurs when ML models fail or behave unpredictably due to unforeseen changes or issues in their data sources that are not easily traceable. The framework proposed uses an analysis that constructs a control flow graphs to automatically detect and manage these dependencies, ensuring that any changes in data sources are quickly identified and mitigated before they can negatively impact the model’s performance. They provide use cases being a dashboard for MLOps engineers to monitor the dependencies, a constructed knowledge graph linking activities and their data dependencies, and in feature store management when multiple models use the same data. They show that their dependency mapping construction with activity graph approach is able to maintain dependencies up-to date and is comparable to prior work that uses python scripts. This is significant for AI engineering as it addresses a critical bottleneck in deploying and maintaining robust ML systems at scale, where data issues can be more disruptive than code issues. This paper’s approach to automating the detection and management of data dependencies could potentially be applied to safeguard the data pipelines feeding into MAB algorithms in clinical settings, ensuring consistent and accurate data flow and reliable performance of the algorithms. In a larger AI-intensive software project, such as a comprehensive healthcare management system incorporating personalized treatment strategies, the ideas from this paper would be invaluable to maintain the integrity of the data sources feeding into various predictive models. The automated dependency mapping framework could be integrated to monitor and manage the complex data dependencies that support the personalisation algorithms. By ensuring that changes in patient data or other relevant datasets do not disrupt the operation of these algorithms, the framework would help maintain the reliability and effectiveness of the personalized treatment recommendations.

If my research output was directly deployed in clinical settings, my workflow could be adapted to incorporate mechanisms that anticipate potential data source issues flagged by the automated dependency framework. For instance, parts of the system like LVM estimation could be changed to include contingency plans or fallback strategies in cases where certain data sources become temporarily unreliable or inconsistent. This would make the treatment personalization process more resilient to data source disruptions.

References

1. Kinyanjui, Newton Mwai, Emil Carlsson, and Fredrik D. Johansson. "Fast treatment personalization with latent bandits in fixed-confidence pure exploration." *Transactions on Machine Learning Research* (2023).
2. Mwai, Newton, Milad Malekipirbazari, and Fredrik D. Johansson. "Batched fixed-confidence pure exploration for bandits with switching constraints." *ICML 2024 Workshop: Aligning Reinforcement Learning Experimentalists and Theorists*.
3. Paleyes, Andrei, Han-Bo Li, and Neil D. Lawrence. "Can causality accelerate experimentation in software systems?." *Proceedings of the IEEE/ACM 3rd International Conference on AI Engineering-Software Engineering for AI*. 2024.
4. Boué, Laurent, Pratap Kunireddy, and Pavle Subotić. "Automatically Resolving Data Source Dependency Hell in Large Scale Data Science Projects." *2023 IEEE/ACM 2nd International Conference on AI Engineering–Software Engineering for AI (CAIN)*. IEEE, 2023.