WASP Software Engineering Assignment

Ross Irwin
Chalmers University and AstraZeneca

August 2025

1 Introduction

Generating small molecules which bind to protein targets is a crucial part of designing new medicines to treat diseases such as cancer, diabetes, heart disease, and many others. Current approaches to molecular design and optimisation involve a process of large-scale trial-and-error. This often means running many lab experiments to collect information on how well each molecule is performing, and then making changes to the molecules based on this data, and repeating more experiments in a loop. This process is very expensive and time-consuming, and, due to human assumptions made during the design phase, often leads to molecules which don't perform well in human clinical trials.

My research is focused on applying artificial intelligence to design better molecules, faster. Specifically, I have worked on using generative AI techniques such as diffusion and flow matching to improve the speed and performance of AI models for 3D molecular generation – models which design both the molecule and its 3D structure simultaneously [5]. I have also applied a similar framework to train models to design molecules which fit inside a protein pocket [3], ie. the position within the protein they would bind to. This allows the model to generate molecules with size, shape and electrostatic features which are complementary to the protein pocket they are designed for.

Since molecular design and optimisation is traditionally a time-consuming process involving many experimental iterations, the overall goal of molecular design AI models is to leverage existing data in order to cut the time required to design high-performing molecules. For a generative model such as those I described above, this could mean learning how existing drug molecules bind to a particular protein and making use of this information to design new molecules. The shape complementarity and interactions formed between the small molecule and protein are a crucial aspect of the strength of binding. A generative model can exploit these features, along with many others, to design molecules with strong binding affinity for a particular protein in a way that would take humans much longer to unravel.

2 Lecture Principles

Curating Test Sets In Lecture 2 Robert discusses how it's important to pay attention to how validation and test dataset splits for machine learning models are curated in order to better understand the performance of the model on different subsets of the data. He gives the example of an IMDB sentiment analysis tool which at first glance performs very accurately (accuracy > 0.9) but performance drops significantly when only evaluating the model on older movies (accuracy of 0.79 for movies from 1910s) [1]. Since the validation data is likely to highly biased towards more recent films, the headline accuracy figure can give the wrong impression.

Similar results have also recently been reported for 3D molecular design and co-folding tools [2, 7]. These papers have found that docking and co-folding models, which aim to generate the joint 3D structure of protein with a bound small molecule, perform much better on proteins that are structurally very similar to those in the training data. When the same models are tested on protein structures that is less well-represented in the training set their performance drops dramatically. On our recent paper we made use of these findings and curated a dataset which showed distinct structural differences between training and test splits, and showed that our model was more robust to this distribution shift that previous approaches [3].

ML Technology Readiness Level The Machine Learning Technology Readiness Level described in Lecture 4 outlines a framework which helps to describe the level of maturity of a given machine learning system. Since I work as an industrial PhD student at AstraZeneca this framework is particularly useful to keep in mind. Most of my work is focused on prototyping and early productionalisation stages of the framework, levels 0 - 4. When some of these research prototypes turn out to be successful ideas, however, they are often fully productionalised and put into real-world use. Indeed, a number of my previous research projects are currently being used in live drug design programmes, and I have been able to be a part of the productionalisation and deployment process. Particularly important part of the process in these cases were understanding the key user requirements, as well as improving the maturity of the model to ensure more robustness to unusual inputs and distribution shifts.

3 Guest Lecture Principles

Stakeholder Elicitation In his guest lecture, Julian Frattini mentioned that one of the key first steps for performing requirements engineering is to perform stakeholder elicitation. This refers to the process of listing all relevant stakeholders for a project, and outlining the relationships between different stakeholders. Since my research projects are often exploratory in nature the stakeholders to begin with are usually just me, my supervisors, and my collaborators. However, when these models are deployed or published, the stakeholder base widens to

include potential users of these products including other researchers, computational or medicinal chemists, and projects managers within my department. The relationships between these stakeholders can be very complex and would require a much longer essay to discuss in depth; at a high level, however, I think these stakeholders can grouped into builders, users and managers. Builders, like me and other engineers in my department, are tasked with developing and deploying products which will be used by the users. These users, such as medicinal and computational chemists, run the tools on their specific drug projects. Managers oversee all of the processes and set goals for both the builders and users.

Requirements Elicitation Another key element of the requirements engineering process, as outlined by Julian Frattini, is the requirements elicitation step. This involves working with end users to decompose system goals into specific and measurable requirements. For me, in industry, these requirements often come from features requested by users through business reference or user groups. These groups contain a diverse subset of end of users of products and they collectively work, alongside engineers, to break feature requests down into specific requirements that engineers will then develop in their products.

4 Data Scientists vs Software Engineers

Chapters 1 and 2 of the book "Machine Learning in Production" [6] discuss some of the difference between data scientists and software engineers, as well as providing some opinions on how these roles may evolve over time. I mostly agree with the sentiment outlined in these chapters, that engineering and data science are two distinct roles and each role tackles different types of problems and requires different mental models for approaching these problems. However, I think the book at times provides an overly simplistic view of these two roles; while it is true that researchers often don't directly work on configuring MLOps infrastructure, researchers do spend a lot of time working on engineering tasks. I have personally found that building a platform on which research-type experiments can be run usually takes much longer than running the experiments themselves.

I believe engineering and research are and will continue to be fundamentally different roles, but roles which require a lot of common knowledge and skills. As the book suggests, I also believe that ML engineering will become a much more common role, and will be widely seen as distinct from ML research. I'm basing this belief on the observation that the complexity of these systems is growing dramatically, and will likely continue to grow. With the added complexity of these systems lots of different expertise will be required - recent papers introducing new LLMs often have 10s or even 100s of authors. I believe we may even start to see more specialisation within research and engineering roles, for example seeing a pre-training researcher as distinct from an RL researcher.

5 Paper Analysis

5.1 Paper 1

The first paper I picked is "Code Smells for Machine Learning Applications" [8]. This paper outlines a list of 22 ML-specific code "smells" which were identified from a variety of sources. The key idea of the paper is that ML-specific code contains specific fingerprints and patterns which can lead to quality problems when no structures are in place to deal with them. These ML-specific smells go beyond patterns that have been previously identified in traditional software engineering projects. These ML-specific patterns are important for ML researchers and engineers to understand since they can lead to error-prone code or poor performance and efficiency of ML processing pipelines.

Since my research often deals with very large molecular datasets (these can be over one billion molecules), efficient data processing pipelines are crucial for working at large scale. Understanding code smells identified in this paper would help researchers such as myself to reduce errors and improve the efficiency of my code. Examples of smells the authors identified which would apply in this case include not using vectorised operations, include redundant loop iterations, and incorrectly initialising Pandas dataframes. Another code smell identified by the authors which is particularly important to my work is data leakage. As I discussed above, data leakage, which can be either explicit (by duplicating data between train and test sets) or implicit (by allowing structurally similar molecules to appear in both sets), has been identified as a significant problem when evaluating molecular generation and co-folding models.

One of the major molecular generation platforms which I am involved with inside AstraZeneca is called Reinvent. It uses a chemical language model along with reinforcement learning to generate drug-like molecules which optimise a user-defined scoring function. Since this platform is required to deal with a large number of molecules a number of the ML code smells identified in the paper would be useful to further investigate possible inefficiencies in this code. Some of smells identified also relate to the reproducibility of ML models, such as not setting deterministic flags or fixing random seeds. Further research into the Reinvent platform could investigate how reproducible current models are and whether forcing determinism in the model may allow for more thorough evaluation. Much of my research directly relates to the prior models on the Reinvent platform, although my work is still in a prototype phase and is just beginning to be deployed to end users. Many of the smells identified in the paper are not related to my research, but some, particularly those relating to potential inefficiencies, may have an impact on my work.

In the future I intend to pay better attention to ML-specific antipatterns and code smells in my projects. Ultimately, what this paper suggests is that the key understanding common mistakes made in ML processing and model code and then being able to identify and avoid those mistakes in the future. For me specifically, this means looking out for some of the patterns they have identified, both at the initial research stage of the project, but also particularly when the

project is deployed to end users since this is where errors in the code could have more significant detrimental impact.

5.2 Paper 2

The second paper I read was "Data Smells: Categories, Causes and Consequences, and Detection of Suspicious Data in AI-based Systems" [4]. Rather than dealing with code smells, this paper investigates data smells relating to machine learning projects and datasets. Specifically, they investigate and catalogue a set of potentially unwanted patterns found in data used to train machine learning systems. At a high-level they categorise smells into: believability smells - implausible data values, understandability smells - inappropriate or ambiguous values, and consistency smells - data columns with inconsistently labelled values. Understanding these issues and identifying these smells in ML datasets is important for AI systems since poor quality data can significantly affect the performance of AI systems. The patterns this paper has outlined can help AI engineers and researchers to identify questionable data or data which may cause further problems downstream within their datasets.

While many of the identified smells do not have an exact equivalent within molecular data, some of the data smells in the paper are also quite relevant for molecular machine learning datasets. Tabular data is particularly common for datasets relating to molecular property and binding affinity prediction. I have witnessed first-hand a number of the data smells identified within the paper in these molecular datasets, particularly issues such as dummy values, missing values, and suspicious values. Since these data are derived from real wet-lab experiments they are inherently noisy and prone to missing values. In these cases experimentalists sometimes fill in missing data with dummy values, which can appear to be suspiciously high or low, based on the other data. Additionally, like other tabular datasets, molecular data can also contain consistency smells such as providing different binding affinity measurements (eg. IC50, K_i or K_d) for different molecules within the same data column, or providing different units (eg. nanomolar or micromolar) within the same column.

Reusing the example of the Reinvent platform from above, there are a number of ways this paper's findings could influence the development of the platform. Reinvent uses scoring functions, such as estimated binding affinity, solubility or logP, to help guide the search through chemical space. The goal of the system is then to generate molecules which optimise a given combination of scoring components. These scoring components are often trained on tabular data which, as mentioned above, can contain questionable data values. Using the catalogue of data smells described in the paper could help to better understand data quality issues associated with these datasets and help inform improved data filtering and cleaning efforts, which will ultimately lead to training of better models.

While the paper only deals with tabular data, and my research has so far mostly focused on generating molecular structures, there are a number of places where the findings of this paper could help with my future work. Firstly, it is likely that in the near future I will be working on training predictive models

for properties such as binding affinity, and here, as described above, many data quality issues persist. Understanding the data smells present in these datasets will help to improve the filtering and cleaning of them. Secondly, while tabular data smells are not directly applicable to datasets of molecular structures, the same idea could be applied to molecule datasets in order to improve the quality of the generative models trained on them. For example, datasets of molecular can be very large and likely contain many molecules which are not desirable for generative models. Identifying some data smells such as undesirable molecules or data bias, could help to train improved generative models in the future.

6 Research Ethics and Synthesis Reflections

Search Process For each CAIN conference from 2022 – 2025, I looked through the list of all accepted papers at that conference and then created a shortlist of papers I found interesting from their titles and abstracts. I then looked into the shortlisted papers in a bit more detail, for example, reading the abstract (if not already) or the key findings from the paper. I then chose the two papers outlined above as I thought they were both very applicable to either my research or to the work we do in AstraZeneca. I think analysing these two papers together was also particularly interesting because they both deal with the same phenomenon but in different sources - one in code and one in data.

Pitfalls and Mitigations Creating a shortlist of papers and then applying an extra filtering step by looking into the key findings of the paper helped to mitigate investigating papers with misleading titles. With this strategy I was able to find the above two papers fairly quickly and found that both were applicable to my work.

Ethical Considerations I cited all relevant sources (outside the lectures) while rewording their findings into my own writing, and did not copy from any LLM.

References

[1] Guy Barash, Eitan Farchi, Ilan Jayaraman, Orna Raz, Rachel Tzoref-Brill, and Marcel Zalmanovici. Bridging the gap between ml solutions and their business requirements using feature interactions. In *Proceedings of the 2019 27th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, pages 1048–1058, 2019.

[2] Martin Buttenschoen, Garrett M Morris, and Charlotte M Deane. Posebusters: Ai-based docking methods fail to generate physically valid poses or generalise to novel sequences. *Chemical Science*, 15(9):3130–3139, 2024.

[3] Julian Cremer, Ross Irwin, Alessandro Tibo, Jon Paul Janet, Simon Olsson, and Djork-Arné Clevert. Flowr: Flow matching for structure-aware de novo, interaction-and fragment-based ligand generation. *arXiv preprint arXiv:2504.10564*, 2025.

[4] Harald Foidl, Michael Felderer, and Rudolf Ramler. Data smells: Categories, causes and consequences, and detection of suspicious data in ai-based systems, 2022.

[5] Ross Irwin, Alessandro Tibo, Jon Paul Janet, and Simon Olsson. Semlaflow – efficient 3d molecular generation with latent attention and equivariant flow matching. In *International Conference on Artificial Intelligence and Statistics*, pages 3772–3780. PMLR, 2025.

[6] Christian Kästner. *Machine Learning in Production*. MIT Press, 2025.

[7] Peter Škrinjar, Jérôme Eberhardt, Janani Durairaj, and Torsten Schwede. Have protein-ligand co-folding methods moved beyond memorisation? *BioRxiv*, pages 2025–02, 2025.

[8] Haiyin Zhang, Luís Cruz, and Arie van Deursen. Code smells for machine learning applications, 2022.