Assignment, WASP Software Engineering

Maja Lindström

August 6, 2025

1 Introduction

I am an industrial PhD student at Siftlab AB and Umeå University. I am doing research in network science which is an interdisciplinary field closely related to graph theory. Network science focuses on understanding complex systems through studying connections and relationships. My most recent work is on community detection, particularly identifying overlapping communities (groups of nodes that may belong to multiple communities simultaneously) [1]. To detect these communities, I use the map equation framework [2, 3, 4], which is a method based on information theory. It models flows on networks using random walks and identifies communities by compressing the description of this flow. The idea is that good communities allow for a more efficient (shorter) description of movement within the network.

I have also been working quite a lot with link prediction in networks. This means finding missing or future connections in a network. This is especially challenging and important in sparse networks, where observed data is limited or links have not been formed yet. Applications are for example biological networks, such as predicting unknown protein-protein interactions, and recommender systems, where the goal is to suggest likely but unobserved interactions between users and items [5].

2 Lecture Principles

I think the main thing I will take away from this course is to think more before doing and with that I mean to invest more time into planning and understanding the problem before jumping into implementation. In my research, the projects often become exploratory and open-ended which makes planning, structure and understanding even more important. I now try to take time to design experiments, define objectives, and map out data flow which improves the clarity and also the impact of the research. It also makes the codebase easier to maintain and extend which is important when collaborating with others or revisiting the project months/years later.

Another important principle that I bring with me is to do testing throughout the project, especially unit testing. Before this course, testing throughout the project has often been overlooked in the rush to get results. But I have learned that doing unit tests is important to find bugs early and improves reproducibility by a lot. I collaborate with other people and to have a well-working code enables them to contribute safely without accidentally breaking things or similar.

These practices of planning ahead and testing continuously are not just good engineering, they also makes it more reproducible, and easier for collaborative research. They help turn experimental code into reusable, transparent tools that others can build on.

3 Guest-Lecture Principles

Requirements Engineering: I am an industrial PhD student at Siftlab, an e-commerce company that helps retailers through data analysis and recommendation systems which means I work at the intersection of academia and industry. This creates a complex environment with multiple stakeholders: Siftlab as a company, its diverse clients, the university, my academic supervisors, and myself. Balancing and managing the expectations of all these parties can sometimes be challenging.

The guest lecture on requirements engineering has helped me think more systematically about how to handle these competing demands. In particular I found the importance of

eliciting, specifying, and documenting requirements clearly very interesting and something I took with me. I now try to write down and formalize requirements (both functional and non-functional) in a way that is understandable and agreed upon by all involved. This clarity helps prevent misunderstandings, especially when trying to navigate between academic goals and commercial priorities.

Another insight was the need to account for different types of stakeholders, each with unique goals and constraints. For instance, some of Siftlab's clients prioritize real-time performance, while others care more about interpretability or predictive accuracy. This variety forces me to consider not just one ideal solution, but a flexible framework that can be adapted to different needs. The lecture reminded me that requirements engineering is not just about technical specs, it is more about communication and collaboration across different perspectives.

4 Data Scientists versus Software Engineers

I do agree with the essential differences between data scientists and software engineers from the CMU "Machine Learning in Production" [6] chapters since I personally relate more to the data scientist profile and less to the software engineering one. I think these roles differ in educational background, mindset, and typical workflows as explained in the book. They also mention in the introduction that this is very oversimplified and overgeneralized which I believe is true but I agree with them the broader perspective. However, I would add that I think the landscape is shifting. Some data scientists are now more actively involved in deploying models (especially in smaller teams), and many software engineers are gaining competence in ML frameworks. In education, I can also see a shift as courses are becoming broader and more interdisciplinary. So while the divide between the roles still exists, it is not always as rigid as the book describes.

I do not see that we will get a full merge of the two roles. I believe that people are driven by different interests, and their strengths often align more with one discipline than the other. I believe we are moving toward the T-shaped skill set discussed in the book, specialists in one domain (either data science or software engineering) with a working understanding of the other. This does not mean every data scientist must become an expert in Kubernetes, or every software engineer must master gradient boosting, but having enough background to understand trade-offs, constraints, and interface expectations will be important.

5 Paper Analysis

5.1 Generating and Verifying Synthetic Datasets with Requirements Engineering

5.1.1 Core ideas and their SE importance

The first paper I chose to analyze is the paper 'Generating and Verifying Synthetic Datasets with Requirements Engineering' [7]. This paper presents a way to generate synthetic image data using diffusion models (like Flux), but in a structured and reliable way by applying requirements engineering. Instead of writing random prompts, the authors define clear requirement specifications (like in traditional software engineering) to describe what kind of images are needed. These specifications are then used to guide the image generation process and to verify whether the generated images meet their desired criteria.

To check if the images are good enough, they use a combination of manual inspection and an object detection model (YOLOv8). This model simulates the behavior of a future machine learning system that will be trained on the images. If both the human and the model agree that the image meets the requirements, it is accepted, and otherwise, it is rejected.

This is important for software engineering because it brings structure, traceability, and verification into the creation of data for AI systems. It makes the process more transparent and trustworthy. It is for example very useful when using AI in safety-critical or high-risk applications. It also bridges the gap between software engineering and machine learning development by treating training data as part of the system that must be tested and verified.

5.1.2 Relation to My Research

This paper relates to my research in many ways but what I liked most is its focus on generating and verifying synthetic data in a structured way. In my case, I work with networks and use link prediction to understand or forecast relationships between nodes. Often, getting enough labeled or realistic network data is challenging, similar to the lack of image data discussed in the paper.

Another thing that I enjoyed with this paper is how they use diffusion models to generate synthetic images with clear requirement specifications. Instead of random prompts, they describe exactly what kind of object should be in the image, where it should appear, and how many times it should appear. This controlled generation makes the dataset more useful and predictable. They also verify the images with a pre-trained object detector to make sure the synthetic data supports the behavior they want from the model.

This idea could inspire something similar in my research. I could explore generating synthetic network data, for example graphs with specific properties or communities using graph generation models guided by specific structural requirements. For example, if I want to study how well my link prediction model performs in sparse vs. dense networks, I could define those properties in advance and generate synthetic graphs accordingly. Just like the paper uses object detection to verify the data, I could use link prediction results or graph metrics to check if the generated data meets my goals.

5.1.3 Integration into a larger AI-intensive project

I am thinking a project like an autonomous public transport system that uses both visual perception and social behavior modeling to navigate through cities. The visual part would use cameras and object detectors to identify people, vehicles, and obstacles. The social modeling part could use network science to predict movement patterns, for example where people tend to gather or how groups move through an area.

In this system, high-quality training data is important for both components. The method from the paper could be used to generate synthetic images showing specific traffic or pedestrian situations that might not be well represented in real data. These images would follow written specifications, for example 'a bus on the left side of the image with three people nearby', and be verified using a model like YOLOv8. This makes sure the visual system learns to detect important scenes, even if they are rare in real life.

My research on link prediction could support the social modeling part. I could train models to predict how people will move or interact based on network data. If real interaction data is limited, I could also explore generating synthetic social networks that follow specific behavioral rules and patterns. These could be validated in a similar way for example by checking if a link prediction model behaves as expected on the generated data.

Together, both the synthetic image generation method from the paper and my research on networks could help build a more complete and well-tested AI system. Using requirements engineering for both parts would make the system more transparent, reliable, and easier to validate.

5.1.4 Adaptation of My Research

I could adapt my research by thinking more about how synthetic data is created and verified in the context of networks. Right now, my focus is mainly on using real-world network data or standard datasets for link prediction. But this paper shows that it can be useful and efficient to generate my own data.

I could include a synthetic network generation step, where I define specific properties or patterns I want the networks to have. For example, I might require that certain nodes act as hubs, or that communities overlap in certain ways. These specifications could be used in a network generation model, like the image prompts in the diffusion model in the paper.

I could also introduce a verification step to check if the synthetic graphs really meet those requirements. This would make my link prediction pipeline more structured and easier to test.

5.2 Investigating the Impact of SOLID Design Principles on Machine Learning Code Understanding

5.2.1 Core ideas and their SE importance

The second paper I chose to analyze is the paper 'Investigating the Impact of SOLID Design Principles on Machine Learning Code Understanding' [8]. In this paper, they investigated if applying the SOLID design principles (which were originally created for traditional object-oriented software) could improve how good data scientists understand machine learning code. The authors did a study containing 100 participants from both academia and industry. One group was shown raw, unstructured ML code from a real industrial system, while the other group saw the same code restructured using SOLID principles.

The results showed that the SOLID version of the code made it much easier to understand and maintain, especially for professionals. Participants liked that the responsibilities were nicely separated and that the code was easier to extend and change without breaking existing parts.

This is important because many ML projects are built by people with limited software engineering backgrounds, leading to messy or hard-to-maintain code. By showing software engineering best practices like SOLID can help in ML settings, this paper strives after stronger integration between ML development and SE principles, which is important for creating sustainable and production-ready AI systems.

5.2.2 Relation to My Research

I work with machine learning models within network science and link prediction which is different to this paper's topics, however, the principles can still be used. ML models can become messy and hard to maintain and use (especially when collaborating across different disciplines and different people with varying knowledge). The paper shows that applying design principles improves code clarity and maintainability, which I could benefit from in my own research. For example, I could use SOLID principles to separate graph data loading, feature extraction, model training, and evaluation into more clearly defined components. This would help me (and others who read or reuse my code) to understand and extend it more easily.

5.2.3 Integration into a larger AI-intensive project

Let us think simple and think of a large AI-based system at some big company. It could for example be a platform designed to help data scientists, software engineers, and machine learning engineers build, test, and deploy ML models across different domains, such as finance, healthcare, or e-commerce. The system includes tools for data preprocessing, model selection, hyperparameter tuning, and monitoring performance after deployment, for example. This platform has many contributors, some with a software engineering background and others coming more from data science. As the platform grows, the codebase becomes harder to maintain. Without clear structure, different parts of the system become entangled which makes it difficult to track changes, debug issues, or test new ideas.

The ideas from the paper could improve the maintainance of this platform. Applying SOLID principles would help modularize the ML pipeline (for example, separating data loading, model training, evaluation, and deployment into well-defined components). This would make it easier for different contributors to work on different parts of the system without stepping on each other's toes. It would also allow easier experimentation with new models or evaluation metrics without breaking existing functionality.

My own research, which is on graph-based ML techniques like link prediction, could be one of the plug-in modules within the platform. Using SOLID principles would help make my component more reusable. For example, defining a clean interface for graph-based predictors, could allow the system to switch between different prediction methods (random forests, GNNs, flow-based methods) with just small changes to the main code.

5.2.4 Adaptation of My Research

To bring with me the ideas in this paper, I could update my research project to follow SOLID principles more explicitly. Currently, my code is mixing data loading, graph building, and model evaluation in the same notebooks. This makes it harder to reuse parts of the pipeline

or swap out components. These changes would make it easier to share my work, scale it up, or apply it in industry settings where software quality and maintainability may matter even more. This aligns with the paper's message: ML development should follow solid software engineering practices to be sustainable and robust.

6 Research Ethics & Synthesis Reflection

I wanted to select newer papers since this field evolves quickly, and I expected that more recent work would better reflect current challenges and practices. I started my search by looking through the 2025 proceedings of the CAIN conference and selected one of the papers from there. Then I moved to the 2024 proceedings and selected one paper from there.

The screening process: I first read titles and filtered there first. Then I read the abstracts of some of the papers where I found the title interesting and that seemed relevant to the course. I particularly looked for papers that discussed concrete engineering practices, like design principles or data pipelines.

One challenge I encountered was that some titles and abstracts sounded relevant but turned out to be too theoretical or too focused on narrow technical implementations without a broader SE perspective. Therefore, I prioritized papers that described empirical studies, real systems, or engineering methodologies instead since these aligned more with the course.

To ensure originality, I tried to take my own notes from the papers and always tried to think how this could be used in my own research. I did not just copy from LLM sources, all ideas are mine.

References

[1] Maja Lindström, Rohit Sahasrabuddhe, Anton Holmgren, Christopher Blöcker, Daniel Edler, and Martin Rosvall. Mapping compact memory-biased dynamics reveals overlapping communities, 2025.

[2] Martin Rosvall and Carl T Bergstrom. Maps of random walks on complex networks reveal community structure. *PNAS*, 105(4):1118–1123, 2008.

[3] Martin Rosvall, Daniel Axelsson, and Carl T Bergstrom. The map equation. *Eur. Phys. J. Special Topics*, 178(1):13–23, 2009.

[4] Daniel Edler, Anton Holmgren, and Martin Rosvall. The MapEquation software package, 10 2022.

[5] Maja Lindström, Christopher Blöcker, Tommy Löfstedt, and Martin Rosvall. Compressing regularized dynamics improves link prediction with the map equation in sparse networks. *Phys. Rev. E*, 111:054314, May 2025.

[6] Christian Kästner. *Machine Learning in Production: From Models to Products.* MIT Press, Cambridge, MA, 2025. Published April 8, 2025. All author royalties are donated to Evidence Action. Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International Public License. Corresponds to CMU course 17-645 / 11-695.

[7] Lynn Vonderhaar, Timothy Elvira, and Omar Ochoa. Generating and verifying synthetic datasets with requirements engineering. In *2025 IEEE/ACM 4th International Conference on AI Engineering – Software Engineering for AI (CAIN)*, pages 212–221, 2025.

[8] Raphael Cabral, Marcos Kalinowski, Maria Teresa Baldassarre, Hugo Villamizar, Tatiana Escovedo, and Hélio Lopes. Investigating the impact of solid design principles on machine learning code understanding. In *Proceedings of the IEEE/ACM 3rd International Conference on AI Engineering-Software Engineering for AI*, pages 7–17, 2024.