WASP Software Engineering Assignment

Nithesh Chandher Karthikeyan
nithesh.chandher.karthikeyan@liu.se
Linköping University

August 2024

1 Research Topic

My research is focused on developing methods in deep generative models to generate synthetic images of high quality and photo-realism, that can be used for real-world applications. Further, I would like to extend the existing pipeline for more controllable image generation based on user inputs. I intend to explain my research project briefly as follows:

In recent times, the large scale generative models for generating images like Dall-E 3 [1], Stable Diffusion [6], Imagen [7], Midjourney etc., are used as foundational models, which are trained on vast amount of data and are capable of producing visually pleasing, photo-realistic images. The question of whether such models can be utilized to produce synthetic training datasets arises since gathering real-world data is a challenging and time-consuming process. At present, I am working on generating synthetic dataset for training deep learning models by leveraging the embedding space of the diffusion models. The proposed method is called as *Diffusion Inversion* as it shares similarity with GAN inversion [9], but we aim to invert a given image back to the conditioning space of the pre-trained diffusion model. Once these images are inverted into the conditioning space, multiple synthetic images can be sampled with similar features to the original images by perturbing the corresponding conditional vectors. The method is being evaluated on multiple benchmark dataset and has so far shown some promising results but it still faces minor issues like generalization problem during model training. As part of the future works, I am planning to extend this framework in real-world application like medical imaging, computer vision use-cases and many more.

From my project outline, we can see that it is part of the Machine Learning core, as the objectives are mostly based on working and developing methods in deep generative models. However, the end goal of the project is about building a framework/ platform for synthetic dataset generation which calls for software engineering elements like reproducability, GPU-optimized implementation, error detection/diagnosis, and testing frameworks, as well as front-end and back-end components like UI, and API facilities if needed.

2 Course Concepts Related to My Research

Despite my lack of prior experience in software engineering, Robert's lectures provided me with a variety of interesting concepts that I found relevant to my research. Some of these concepts I have already applied, and some I may apply in the future.

2.1 Verification and Validation

The concept of *Verification and Validation* is the most relevant to my research among the topics covered in Robert's lecture. Verification is the process of examining whether the project or software generates the desired output that was expected based on the project requirements, specifications and implementation. In the context of my research, it's important to examine the conditioning vectors generated by the model

for a given set of input images and whether it is possible to construct meaningful synthetic images using the learned vectors. Examination is done by verifying different components of my project like whether the training setup meets with project requirements, whether the training's loss curves produce the expected results, and above all, analyzing whether the synthetic dataset generated is comparable to the original dataset. Most simplest verification that I usually do for the above cases is to define automated unit test for different components of the source code and check whether it satisfies those test cases. Although these methods can help my implementation to generate the desired synthetic dataset, there is no guarantee that the given model can be used for real-world datasets. This issue can be solved through validation, where the model is tested on multiple real-world datasets (including benchmark datasets). Also, comparisons of real and synthetic datasets are always a problem as it is perceptual evaluation and the evaluation metrics like Frechet Inception Distance (FID)[4], Kernel Inception Distance (KID)[2] and Inception score are unreliable in some cases. In such situations, the datasets can be evaluated based on the application by implementing a corresponding downstream task as a test cases (For example: If the application is image classification, a model can be trained separately on Real and Synthetic datasets, and the performance of these models corresponds to the quality of the datasets).

2.2 Group Level BSE

Behavioural software engineering (BSE) is another interesting idea from Robert's talk that I can use for my research. BSE, to explain it briefly, is the understanding of a person's, a group's, or an organization's psychology in relation to software engineering activities. While BSE has various components, Group Level BSE has more relevance to my research. During my PhD, I will be working on multiple collaborations within my research group and with WASP NEST environment. Working on a collaborative project is completely different compared to my present individual project as concepts like group dynamics must be taken into account. This may have to do with how comfortable the group members are within the group while interacting, sharing creative ideas, assuming roles, making decisions, being productive, inclusive, etc. Additionally, there can be situations with practical difficulties and disagreements in a group. During my recent collaboration, we faced problems, like how to move forward in an event with contradictory ideas and minor issues like how to contribute to shared git repo without conflicts as there were too many commits to the main branch. Therefore, for a successful collaboration, it is important to pay attention to Group Level BSE.

3 Guest Lecture Concepts Related to My Research

Guests lectures given by Per Lenberg from SAAB and Dhasarathy Parthasarathy from Volvo Trucks provided us a broader view on how software engineering and machine learning works in the industries. Some of my insights from these lectures are discussed below:

3.1 SAAB Guest Lecture

Per Lenber gave an overview of Saab's approach to integrating AI to the existing software engineering pipeline. He started by explaining how crucial is software engineering in Saab and the importance of human factors in software engineering through BSE. He shared some intriguing insights about how people in Saab foresee about increased integrating of AI/ML in software engineering. Most of them raised the uncertainty over the impact of AI/ML in Software engineering roles and acknowledges that the fundamentals of software engineering pipeline will still remains unchanged despite the growth in AI/ML. However he concluded by mentioning that despite the reluctance/ uncertainty in the adoption of AI/ML in software engineering systems, it is stll believed that the organization/ hierarchy will invest in AI integration as they don't want to risk to the fear of falling behind others. The main takeaways for me in this lecture is the dynamics of how organisations and groups in a corporate like Saab works together

even though they don't completely agree on certain things like the one discussed above on AI integration in software engineering pipeline.

3.2 Volvo Guest Lecture

The core idea of Dhasarathy Parthasarathy presentation was AI-driven test generation. He suggested on using LLMs for generating test cases to validate different services of the system. This approach is time-efficient as LLMs can generate a large set of test cases for different services and run them simultaneously, where it might require hours of human expertise to derive these test cases manually. Even though, these test cases may not work during real-world settings of autonomous vehicles due to dynamic and unpredictable environment, it is still a better alternative for internal/alpha testing. For these reasons, I find this idea interesting and would like to adapt this method for testing my research project.

He also mentioned about using LLMs to address the structural ambiguity of the code-bases of these services. Though it reduces the cost and labour for finding and resolving these ambiguities, it still feels like a short-time fix. If an error is encountered in one of the services during testing, it can propagate throughout the system by affecting the behaviour of other services. Also, finding the source of the error can add more complexity to the existing system. Further, it is still difficult to figure out how to evaluate and test these LLMs that are used for resolving code-base inconsistency.

4 Papers from the CAIN conference

4.1 What About the Data? A Mapping Study on Data Engineering for AI Systems

Heck P, et al. [3] provides an overview of the existing literature on data engineering for AI systems also called as AI data engineering by means of mapping study. The paper emphasizes on how organizations lack the data infrastructure necessary to support AI modeling, but instead focus more on developing AI systems. The mapping study found 25 relevant papers on data engineering for AI systems based on the several selection criteria used by Kitchenham and Charters [5] followed by citation snowballing and in the end classify these 25 papers based on meta-data, scope, type of data engineering and empirical validation (if any). The paper also formulated four research questions on life-cycle phases, technical solutions, architectures and lesson learned on AI data engineering and tries to find solution from the classified papers. The study concludes by stating that while most papers focus on production pipelines and training models for AI systems, data architecture for AI systems is largely ignored. To help practitioners implement the data engineering system, there is also a great need for data pipelines, architectures, tools, and best practices.

My research has many relevance to the mapping study conducted by the paper. My project is part of the data-centric AI, where my research focus on improving and generating the synthetic data required for the training models, which reflects the core idea of the paper that emphasis on AI data engineering. Further, the end goal of my research project is to make this pipeline end-to-end learnable, where the generative model is trained to generate synthetic dataset, which is then used to train models for downstream tasks and based on the performance, use a feed-back loop to improve better sampling of synthetic data in the trained model space. A similar idea is shared in one of the paper classified under technical solution for AI data engineering, where they propose a platform to simultaneously train and deploy machine learning model by adapting the pipeline to the changes in incoming data.

The adaption of the synthetic image generation pipeline to different domain is one of the unexplored areas of my current project. One of the paper in the mapping study introduced a framework for generating synthetic aerial data for landing pad detection in a simulation scene. Although the proposed framework is very specific to the application, it will be interesting to see how generative model can be incorporated in the framework for generating synthetic aerial simulation data. Another interesting topic from the paper that can be utilized in my project is the data validation system in production. Many papers focus on data engineering for training data but AI systems become vulnerable during production/

deployment due to insufficient validation in real-world environment. This requires background knowledge on how to validate systems during production and how to engineer data production pipeline. It will be interesting to think along how generative models can be used to generate the validation data based on the few examples encountered during production and to engineer a real-time production data pipeline.

4.2 An Exploratory Study of Dataset and Model Management in Open Source Machine Learning Applications

Toma T R, et al. [8] discusses about the management of machine learning (ML) artifacts like datasets and models in ML application. The author emphasizes that ML artifacts must be properly versioned and stored, just like software artifacts. In order to support their claim, they carried out an empirical study on 93 selected GitHub ML application repositories with 321 data files and 354 model files. The study focuses on three research questions: In an ML application, how are 1) the data stored?, 2) the models stored? and 3) the datasets and models versioned? The results of the study provided some interesting findings on the common practices of maintaining ML artifacts in the ML applications. The developers prefer to store both datasets and models in their local file systems, making it easier to access while development but not shareable in most of the cases as it resides in their local system. Also, in case of larger file size, the developers use remote storage by loading dataset through URLs but the authors encountered problem with URL-based data access like HTTP 404 (No file found) and HTTP 403 (No permission to access) errors. However, the most interesting finding is that version information is frequently absent from model files and data during application development, which could cause problems for developers with reproducibility and traceability of the application. The authors conclude by mentioning the need for software engineering tools or practices for developers to integrate data and models in ML application.

I can co-relate the findings of the paper to my practice of maintaining the dataset and models in my project. Because of the substantial funding provided by WASP for the Berzelius supercomputer, I use them for training models and store both real and synthetic data, along with model checkpoints, in the Berzelius File System. This practice of storing in the file system is similar to the developer's approach that is discussed in the paper. Additionally, my project development is similar to the work of data scientist, where the main focus is on developing models, finding the right set of hyper-parameters and augmenting dataset (if needed). In contrast, the software development projects tend to focus more on data collection, organizing, versioning, deployment, reproducibility and other related-aspects. Thus, this paper has highlighted some of the shortcomings in managing ML artifacts in my project, which I intend to address in the future.

The paper has also brought out some of the shortcomings in their study. They have restricted the machine learning application in the open source repositories in GitHub but there are also other DevOps platform like Gitlab and Bitbucket, that contains a lot more ML projects that can be added to the study. Also, in recent times, larger models (like LLMs) and datasets are stored in an AI platform called Hugging Face[1] and these resources can be retrieved through simple API calls. Further, the models and datasets are stored separately in their respective repository, simplifying the versioning as the latest commit to each repository corresponds to the newest version. With the establishment of Hugging Face, developers have adopted a new methodology, where the source code of the project are stored in DevOps platform like GitHub and file systems are replaced by Hugging Face. Though the paper have mentioned about not including third-party providers like Hugging Face in their study, it would be interesting to know how this platform has impacted the management of ML artifacts in ML applications.

References

[1] James Betker, Gabriel Goh, Li Jing, Tim Brooks, Jianfeng Wang, Linjie Li, Long Ouyang, Juntang Zhuang, Joyce Lee, Yufei Guo, et al. Improving image generation with better captions. *Computer Science. https://cdn. openai. com/papers/dall-e-3. pdf*, 2(3):8, 2023.

[1] https://huggingface.co/

[2] Mikołaj Bińkowski, Danica J Sutherland, Michael Arbel, and Arthur Gretton. Demystifying mmd gans. *arXiv preprint arXiv:1801.01401*, 2018.

[3] Petra Heck. What about the data? a mapping study on data engineering for ai systems. In *Proceedings of the IEEE/ACM 3rd International Conference on AI Engineering-Software Engineering for AI*, pages 43–52, 2024.

[4] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017.

[5] Staffs Keele et al. Guidelines for performing systematic literature reviews in software engineering. Technical report, Technical report, ver. 2.3 ebse technical report. ebse, 2007.

[6] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.

[7] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in neural information processing systems*, 35:36479–36494, 2022.

[8] Tajkia Rahman Toma and Cor-Paul Bezemer. An exploratory study of dataset and model management in open source machine learning applications. In *Proceedings of the IEEE/ACM 3rd International Conference on AI Engineering-Software Engineering for AI*, pages 64–74, 2024.

[9] Weihao Xia, Yulun Zhang, Yujiu Yang, Jing-Hao Xue, Bolei Zhou, and Ming-Hsuan Yang. Gan inversion: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 45(3):3121–3138, 2022.