WASP Software Engineering and Cloud Computing 2024-Assignment 2

Ziliang Xiong
Department of Electrical Engineering
Linköping University
ziliang.xiong@liu.se

1 Introduction to my research

The advancement in Neural Networks (NNs) largely boosts the perception of autonomous robotics systems. Most research in perception tasks, such as object detection and tracking, aims to improve NNs' accuracy and reduce time complexity, however, the prediction uncertainty is paid less attention. NNs can reach high accuracies and are meanwhile poorly calibrated, which means they are over-confident. This could be dangerous for safety-critical applications in the decision-making process. Access to prediction uncertainty could improve the reliability by human intervention or switching to other perception sensors. For example, camera-based NNs are widely used in autonomous driving. However, they may fail to detect pedestrians in snowy conditions. If the model can return a high uncertainty in such a circumstance, the downstream decision-making unit can choose to rely on other types of sensors, such as radar.

My research lies in the intersection of uncertainty quantification, which is a classic topic in machine learning, and uncertainty-aware perception. The goal of my PhD project is to develop a modular adaptable situation-aware approach for perception so it can enhance the robustness and awareness for situation uncertainty, henceforth improving safety of autonomous systems.

2 Discussion of the relation between my research and topics in the lecture

2.1 AL Software Testing

AI software testing is crucial for my research on uncertainty-aware perception. By generating and selecting diverse test cases, I can expose designed perception systems to a wide range of scenarios, including rare and edge cases, ensuring robustness. Currently, using generative models, such as NerF, to generate test cases gains popularity. Methods like metamorphic testing validate AI systems by checking consistent behavior under different but related inputs, while fuzzing and mutation testing introduce random variations to test the system's response to unexpected inputs. These approaches help identify vulnerabilities and improve the system's ability to handle unforeseen situations.

Additionally, proposing novel coverage criteria ensures comprehensive testing of all aspects of the AI system, including those related to uncertainty quantification. This thorough testing enhances the safety and reliability of autonomous systems, aligning with your goal of improving their robustness and awareness in uncertain environments. By integrating these testing methodologies, I can systematically address the challenges posed by uncertainty and enhance the overall performance and safety of autonomous systems.

37th Conference on Neural Information Processing Systems (NeurIPS 2023).

2.2 AI Software Maintainance

According to the lecture, there are a few open challenges existing in AI software maintainance, such as long-term potential, value of learning, etc,. To address these challanges, it puts an requirement on performance Optimization: continuously monitoring and optimizing the performance of AI systems to ensure they operate at peak efficiency3 and Adaptation and Upgrades: Updating AI models and algorithms to adapt to new data, changing environments, and evolving requirements.

Adaptation to gradually changing domains of data and response to new requirements involves continual learning and online learning. Those are advanced research area. They also affect the robustness of models, e.g., models exposed to out-of-distribution data. My research partly touches these areas, which targets on calibrating uncertainty on those data. The metrics I prosposed can be used in model performance optimization and maintainence.

2.3 Changing with the increased integration of AI/ML

In Per Lenberg's lecture, he reveals that there are different opinions among developers during the operation of SAAB ATM. The top three answers are:

- no more long-term strategies
- Work will be more boring - less development
- Nothing will change - AI is overrated

The first answer is not overly optimistic about the change, reflecting that the industry passively react to the increased integration of AI in software development. From software developer's aspect, the second point is understandable. Of course, any advancement in automation will reduce the manual work. The third answer overlooks that the testing and maintainance of AI-integrated software put a higher demands on developer's AI knowledge. It is comforting that the existing software development process can fit AI-integrated software. However, overlooking that AI models are involving and stochastic might lead to catastrophic failure.

3 CAIN Paper 1:Investigating the Impact of SOLID Design Principles on Machine Learning Code Understanding

Cabral, et al., [1] aim to investigate the impact of the SOLID design principles on ML code understanding. SOLID design principles include

1. Single Responsibility Principle (SRP)
2. Open/Closed Principle (OCP)
3. Liskov Substitution Principle (LSP)
4. Interface Segregation Principle (ISP)
5. Dependency Inversion Principle (DIP)

In summary, these principles guide developers toward writing more maintainable, adaptable, and reliable software. These principles been shown beneficial for understanding and maintainability in traditional software projects. However, ML components are often developed by data scientists with diverse educational backgrounds, leading to code that may not adhere to software design best practices. In this paper, Cabral, et al., [1] conducte a controlled experiment involved 100 data scientists. Real industrial ML code (not using SOLID principles) was restructured to follow these principles. Participants analyzed both the original code and code incorporating SOLID principles. The experiment results show that there is statistically significant evidence suggests that adopting SOLID principles improves code understanding in ML projects.

3.1 Relation to my research

As a doctoral researcher, software engineering and teamwork do not play a central role in my work. This is because of the inherent character of the position and the goal of the project, more specifically,

each PhD student in our division has a unique topic and usually none of our goal is to develop software that is deployable. Teamwork happens sometimes before the deadline of major conferences, however, a common teamwork pattern is that each coauthor is in charge of an independent experiment and report the results to the first author. Therefore, there is no serious need to review each others' code. That means as long as the author can understand his or her own code would suffice. One typical circumstance where readability and maintenance are important is to publish the code, which is usually the last step after a paper is published. In summary, there is no strong relation between this paper and my research but following SOLID principles during programming can enforce better code readability and maintenance.

3.2 A larger Project

My research should fit into an autonomous system software, such as autonomous vehicles (AV). AV software is usually featured with multiple modules functioning in a sequential order. Typical modules include Adaptive Detection and Recognition Framework (ADAF), Vehicle Positioning and Localization (VPL), Path Planning module, Control System, etc..

Although the recent trend is to have the whole pipeline end-to-end (E2E). My research would fit into the ADAF and VPL in the sequential pipeline, whereas it also fits into the E2E pipeline in the way of multitask learning and different task-specific inference head.

SOLID principles in [1] would play a much significant role in an AV software project. In the sequential pipeline, each module's development requires domain experts, hence different teams would develop independently. This development working flow put a high request on code readability and maintenance. Among all the principles, OCP and DIP should be empathised. Between different teams and even between different members within a team, abstractions should be agreed at the very beginning of development, thus interfaces of different modules can be compatible.

3.3 Potential Adaptation

The core algorithm of AV software are usually programmed with C++ and CUDA, however, my research heavily relies on Python and its auto-differential library, Pytorch. Thus, adaptING to SOLID principles in my daily programming will not directly improve the development of an AV software. On the other hand, better readability and reproducibility will alleviate the workload of handover and accelerate the development.

4 A Combinatorial Approach to Hyperparameter Optimization

The core idea of the paper [2] is the application of t-way testing for hyperparameter optimization (HPO) in machine learning. This approach focuses on 't' feature interactions, addressing a key challenge in traditional HPO methods known as the 'curse of dimensionality', which is characterized by an exponential increase in complexity with the addition of each hyperparameter.

This approach is important to the engineering of AI systems because hyperparameters significantly impact the performance of machine learning models. Traditional methods like Grid Search, Random Search, and Bayesian Optimization can be computationally expensive and less efficient. The t-way testing approach proposed in this paper can manage the exponential increase in the number of combinations as the quantity of hyperparameters grows, thus reducing computational expenses and potentially achieving comparable or superior model performance. This makes it a valuable contribution to the field of machine learning and AI system engineering.

4.1 Relation to my research

My research focuses on uncertainty quantification for the safety of autonomous systems. Grid search is the common practice in research of my field to gain optimized model performance over high-dimensional hyperparameter spaces. It requires huge amount of computational resources and adding the waiting time in a queue-based server system. Another tool for managing hyperparameter tunning is visualization programs, such as Tensorboard, Weights& bias. There are usually built in Bayesian optimization tools.

The approach introduced in this paper is particularly useful for managing high-dimensional hyperparameter spaces and detecting faults due to parameter interactions. By focusing on key interactions among hyperparameters, t-way testing ensures thorough exploration without the computational burden of exhaustive search methods, making it a valuable tool for optimizing complex models used in autonomous systems. Integrating t-way testing into your research can significantly improve the reliability and efficiency of your models, which is crucial for ensuring the safety of autonomous systems. It would be even more helpful if T-way testing can be integrated with those visualization tools.

4.2 A larger Project

My research and the core idea of this paper would benefit a larger software project in different aspects.

Integrating the approach from the paper into a larger AI-intensive software project benefits both model selection and model deployments. It can significantly enhance the project's efficiency and performance. This approach can streamline the hyperparameter tuning process, reducing computational costs and improving model accuracy. This method is particularly beneficial for large-scale projects where computational resources are a constraint, ensuring that the models are both efficient and effective.

My research on uncertainty quantification (UQ) for the safety of autonomous systems (WASP) would complement this by providing a critical layer of safety and reliability. Uncertainty quantification is essential for understanding and mitigating risks in autonomous systems, ensuring they operate safely under various conditions. Essentially the evaluation of UQ approachs also relies on metrics, which are parallel to commn performance metrics in this paper. Thus, they should also be possible to optimize by T-way testing.

This synergy between efficient hyperparameter optimization and robust uncertainty quantification would result in a comprehensive solution that not only performs well but also adheres to high safety standards, making it ideal for critical applications like autonomous vehicles or medical robotics.

4.3 Potential Adaptation

Despite the comprehensive experiments in this paper, the model it uses focuses on traditional machine learning models, such as Random forest, XGBoost and shallow neural network. The datasets it uses compared to those in computer vision tasks are rather small. Thus, I am not convince whether the conclusion in this paper can flawlessly hold for deep neural networks, which usually has extremely high-dimensional hyperparameter spaces. As pointed out by the authors, this is a initial work and need further investigation. Given there is no encapsulated visualization tools with T-way testing, I need to implement the pipeline manually and download combinatorial opitmization tool recommended in the paper.

References

[1] Cabral, Raphael, et al. "Investigating the Impact of SOLID Design Principles on Machine Learning Code Understanding." Proceedings of the IEEE/ACM 3rd International Conference on AI Engineering-Software Engineering for AI. 2024.

[2] K. Khadka, J. Chandrasekaran, Y. Lei, R. N. Kacker and D. Richard Kuhn, "A Combinatorial Approach to Hyperparameter Optimization," 2024 IEEE/ACM 3rd International Conference on AI Engineering – Software Engineering for AI (CAIN), Lisbon, Portugal, 2024, pp. 140-149. keywords: Training;Software testing;Fault diagnosis;Software maintenance;Computational modeling;Hyperparameter optimization;Data models;Hyperparameter Optimization;Combinatorial Testing;AutoML,