1. Title: Using ChatGPT to Replace Human in App Testing to Ensure App Usability and Accessibility

Mobile apps now have become the most popular way of accessing the Internet as well as performing daily. Different from traditional desktop applications, mobile apps are typically developed under the time-to-market pressure and facing fierce competition — over 3.8 million Android apps and 2 million iPhone apps are striving to gain users on Google Play and Apple App Store, the two primary mobile app markets. Therefore, for app developers and companies, it is crucial to accelerate the mobile app development process.

Product teams always need to conduct a user study with real and targeted users once the product is developed to test the usability and potential bugs in the products; however, this process is always time-consuming and costly. The team may need to find people of different backgrounds, train them, and then spend time with the users when they are doing the study. Moreover, they always need to conduct several rounds of usability tests every time they iterate the product based on the feedback from the previous study or because of the new requirements from product managers. The emergence of large language models (LLMs) these years reveals a chance to have them replace/assist humans in many different areas [1, 2]. In this topic, we especially focus on ChatGPT, which is a conversational-based LLM that thrives in different fields such as news press, business, programming, and education. We want to explore if ChatGPT has the capability to imitate different end-users (e.g., older adults, children, blind users) to test the usability of products.

Required knowledge:

- Mobile app development
- Strong programming background.
- Basic knowledge about AI/ML and Human-Computer Interaction is a plus.

Note that this project can be carried out remotely. Students with great performance may be granted an opportunity to do a paid Hiwi in the coming semester break and even a PhD position in the future.

[1] Liu Z, Chen C, Wang J, Che X, Huang Y, Hu J, Wang Q. Fill in the blank: Context-aware automated text input generation for mobile gui testing. In2023 IEEE/ACM 45th International Conference on Software Engineering (ICSE) 2023 May 14 (pp. 1355-1367). IEEE.
[2] Feng S, Chen C. Prompting Is All Your Need: Automated Android Bug Replay with Large Language Models. In 2024 IEEE/ACM 46th International Conference on Software Engineering (ICSE)

2. Title: Testing Mobile Apps on Automobile [For both bachelor thesis + practicals]

The integration of mobile applications with automobile systems is rapidly advancing, revolutionizing the driving experience. This bachelor's thesis project focuses on testing existing mobile applications available on Google Play within the unique environment of Android Automotive OS emulators. Different from general mobile apps, car apps are special such as minimizing driver distraction, as any attention taken away from the road can be dangerous, designed for larger, often non-standard screen sizes, subject to more stringent regulatory standards, etc.

Within this project, you will select a diverse range of mobile applications from Google Play that serve automotive-related purposes, such as navigation, entertainment, communication, and other functionalities. Utilize Android Automotive OS emulators to simulate automobile environments and test the selected mobile applications thoroughly. Assess their performance, compatibility, user interface adaptability, and functionality within this specific ecosystem.

This project requires a strong understanding of software testing methodologies, familiarity with Android app development, and the ability to work within emulated environments. Students will gain hands-on experience in testing mobile applications for a specialized platform crucial for the evolving automotive industry. Students participating in this project will not only deepen their knowledge of software testing methodologies but also contribute to the ongoing advancements in the intersection of mobile applications and automotive technology.

Required knowledge

- Strong programming background.
- Experience of mobile app development especially in Android app development.

Note that this project can be carried out remotely. Students with great performance may be granted an opportunity to do a paid Hiwi in the coming semester break and even a PhD position in the future.

https://developer.android.com/training/cars/testing
https://play.google.com/store/apps?device=car
https://developers.google.com/cars/design
https://developer.android.com/docs/quality-guidelines/car-app-quality
https://developers.google.com/cars/design/design-foundations/visual-principles

3. Title: Understanding Deployment Tools for Large Language Models in Real-World Applications: An Empirical Study

Large language models have revolutionized various fields, yet their deployment into real-world applications presents challenges. This bachelor's thesis aims to conduct an empirical study focusing on the issues, challenges, and future prospects of deploying large language models using open-source tools. The project involves a systematic analysis of these tools, their functionalities, and their effectiveness in facilitating the integration of language models into diverse applications.

What you need to do in this project is to:

- Identify and compile a comprehensive list of open-source deployment tools designed for large language models.
- Create a systematic methodology to crawl and analyze issue reports from the repositories of these identified tools.
- Classify and categorize the encountered issues and feature requests based on severity, frequency, and nature.
- Perform a comparative analysis of the tools based on their performance, ease of use, community support, and adaptability to various application types.
- Propose recommendations and potential improvements for the identified tools to address prevalent issues and enhance their usability.

Skills Required:
- Good programming skills, like Python
- Great summarization, presentation and writing capability

Note that this project can be carried out remotely. Students with great performance may be granted an opportunity to do a paid Hiwi in the coming semester break and even a PhD position in the future.

Reference:
https://github.com/Hannibal046/Awesome-LLM?tab=readme-ov-file#deploying-tools
https://github.com/lm-sys/FastChat/issues?q=is%3Aissue+is%3Aclosed

4. Title: Exploring the Role of Large Language Models in Automating User Interface Tasks

In recent years, the integration of Large Language Models (LLMs) has transformed various aspects of technology. One emerging area is the utilization of LLMs to automate User Interface (UI) tasks, ranging from executing complex sequences of actions to manipulating software features across different applications [1, 2]. It can significantly reduce the burden of users, especially the disabled or elderly in using the software. This seminar course aims to delve into the systematic exploration of literature on using LLMs for UI automation.

The primary objective of this seminar project is to conduct a comprehensive and systematic literature review on the application of Large Language Models in automating instructions for User Interface tasks. By reading tens of relevant research papers, students will investigate the capabilities, challenges, advancements, and ethical considerations associated with employing LLMs for executing multifaceted UI tasks.

What you will do will include:

- Develop a method to collect relevant papers.
- Through rigorous analysis, students will synthesize the gathered literature, examining the methodologies, models, algorithms, and applications used in LLM-driven UI automation. They will identify trends, challenges, and potential future directions in this domain.

- Present findings, insights, and conclusions derived from the systematic literature review into a formal research report/paper.
- Create a GitHub repo named Awesome-LLM4UIautomation to display recent relevant research works, tools, etc.

Note that this project can be carried out remotely. Students with great performance may be granted an opportunity to do a paid Hiwi in the coming semester break and even a PhD position in the future.

Reference:
1. Wang B, Li G, Li Y. Enabling conversational interaction with mobile ui using large language models. In Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems 2023 Apr 19 (pp. 1-17).
2. Wen H, Li Y, Liu G, Zhao S, Yu T, Li TJ, Jiang S, Liu Y, Zhang Y, Liu Y. Empowering llm to use smartphone for intelligent task automation. arXiv preprint arXiv:2308.15272. 2023 Aug 29.

5. Title: Aimed at addressing potential intellectual property infringement issues in code generated by large language models

As large language models continue to evolve and more code-assist generation tools based on these models emerge, an increasing number of software developers are using these tools to assist them in software development. However, since the advent of deep learning, intellectual property issues related to code generation by these models have plagued the practical application of technology in industry. Due to the massive amount of data required to train deep learning models to improve their performance, a large amount of open-source but non-commercial code is fed into these models for training. Due to the current black-box nature of deep learning models and their powerful memory capabilities, it is not yet clear whether these models will fully generate code that is non-commercially licensed, potentially leading to intellectual property infringement issues [1,2].

To prevent users from unintentionally replicating complete non-commercially licensed source code while using large models, this study aims to investigate whether there is complete replication of source code from training data in existing large models and propose corresponding solutions to these potential intellectual property infringement behaviors. In this project, students will learn about in-context learning, thought of chains, prompt learning, and other technologies related to large language models, gaining a comprehensive understanding of the advantages and disadvantages of existing large language models in code generation and developing problem-solving skills in related areas.

Required knowledge
- Strong programming background, especially proficient in python.
- Experience of training deep learning models with Pytorch.

Reference
[1] Majdinasab, V., Nikanjam, A., & Khomh, F. (2024). Trained Without My Consent: Detecting Code Inclusion In Language Models Trained on Code. *arXiv preprint arXiv:2402.09299*.

[2] Yang, B., Li, W., Xiang, L., & Li, B. (2023). Towards code watermarking with dual-channel transformations. *arXiv preprint arXiv:2309.00860*.

6.Title: Code Vulnerability Detection with Large Language Models Based on Multi-Agent

Although large language models currently possess powerful code generation capabilities, the security of code generated by these models remains a concern. Current large language models cannot guarantee that the code they generate is free from vulnerabilities. Once this potentially vulnerable code is applied in real software, it could be exploited by hackers to launch attacks, leading to significant economic losses. To enhance the trustworthiness of code generated by large models, it is essential to perform vulnerability detection on the generated code.

Based on existing research, directly using large language models for vulnerability detection has not yielded satisfactory results [1,2]. Therefore, designing effective methods to leverage large language models for efficient vulnerability detection has become a hot research topic.

In this project, students will be required to investigate and summarize existing literature on using large language models for code vulnerability detection. They will also learn about cutting-edge technologies such as multi-agent systems in large language models and apply these techniques to code vulnerability detection.

Required knowledge

- Strong programming background, especially proficient in python.
- Experience of training deep learning models with Pytorch.
- Familiar with static analysis techniques.

Reference

[1] Noever, D. (2023). Can large language models find and fix vulnerable software?. *arXiv preprint arXiv:2308.10345*.
[2] Steenhoek, B., Rahman, M. M., Roy, M. K., Alam, M. S., Barr, E. T., & Le, W. (2024). A Comprehensive Study of the Capabilities of Large Language Models for Vulnerability Detection. *arXiv preprint arXiv:2403.17218*.

7.Title: Query Refinement for Addressing Hallucination Problems in Code Generation with Large Language Models

While large language models currently possess powerful code generation capabilities, researchers have been troubled by the hallucination problem, which hinders further research applications. The hallucination problem refers to the situation where the model-generated content appears reasonable but is actually fabricated and does not meet user requirements. Existing research has also demonstrated widespread illusion problems in code generation by large language models [1]. These illusions may arise due to improper user query formats. Therefore, one approach to addressing the illusion problem is to reconstruct user queries to help large language models understand user needs correctly.

In this project, students will be tasked with exploring the hallucination problems in code generation by existing large language models and investigating effective methods to standardize user inputs to ensure the robustness of code generated by large language models. Through this process, students will gain comprehensive knowledge of the current development status and cutting-edge technologies in large language models.

Required knowledge

- Strong programming background, especially proficient in python.
- Experience of training deep learning models with Pytorch.

Reference

[1] Xu, Z., Jain, S., & Kankanhalli, M. (2024). Hallucination is inevitable: An innate limitation of large language models. *arXiv preprint arXiv:2401.11817*.

Bio: Dr Chunyang Chen is a full professor in the School of Computation, Information and Technology, Technical University of Munich, Germany. His main research interest lies in automated software engineering, especially data-driven mobile app development. Besides, he is also interested in Human-Computer Interaction and software security. He has published 100+ research papers in top venues such as ICSE, FSE, ASE, CHI, CSCW with extensive collaboration with industry, including Google, Microsoft, and Meta. His research has won awards including ACM SIGSOFT Early Career Researcher Award, Facebook Research Award, four ACM SIGSOFT Distinguished Paper Awards (ICSE'23/21/20, ASE'18), and multiple best paper/demo awards. To know more about him, please visit his homepage https://chunyang-chen.github.io/ or official page https://www.professoren.tum.de/en/chen-chunyang