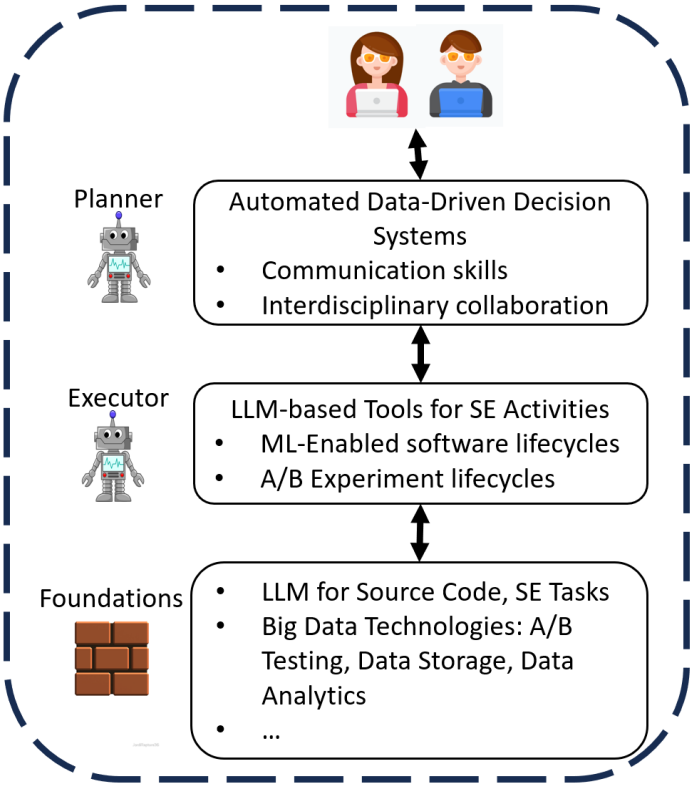
Postdoc Research Proposal: Automated Data-Driven Decision Systems for Software Development

Jie JW Wu
https://jie-jw-wu.github.io/
Aug 28, 2023

We are entering a future where teams of intelligent bots will play a critical role in the software development process. Having worked in both interdisciplinary research topics and industrial software developments, I observe that a proper blend of recent advancements in AI will unlock many exciting future directions in software engineering. My research is driven by a strong desire to **systematically simplify** the complicated, time-consuming, and sometimes tedious software development process. My ambition is to **create automated, data-driven decision systems** that can greatly ease the job of 27 million professional software engineers in the world. Ultimately, this approach aims to **transform the software development process into a series of decisions, supervised by engineers**, who can then focus on more creative things. The outcome of this research proposal will contribute to the exciting future of the **transition from *"Data-Driven Development"* to *"AI-Driven Development"***.



Building *Data-Intensive ML-Enabled Software* Applications

I next summarize my research proposal for the postdoc, assuming a 2-year period.

1. Automated Data-Driven Decision Systems Using Code Language Model

Software industries are shifting from traditional requirement-based development to data-driven development. Companies are embracing a data-driven culture and are adopting data-driven decision-making in software development. **Online A/B testing (a.k.a online controlled experiment, split tests)** and **offline A/B testing (a.k.a offline policy evaluation, offline evaluation)** are two emerging research areas that make informed decisions by comparing two technologies based on online and offline experiments. However, online and offline A/B testing are time-consuming, error-prone, and costly manual activities.

During my doctoral study, I proposed the first approach to formalize data-driven decisions from A/B testing results, using **Multi-Criteria Decision-Making (MCDM)** in a systematic way [5,3]. I formulated the problem of making launch decisions using A/B testing results as a multi-objective optimization. I then proposed an MCDM-based framework to address it. In evaluation, I compared and evaluated 6 Analysis-of-Alternative (AoA) methods and 9 Criteria Weighting methods on a dataset of 5k A/B tests. My result showed that the empirical decision-making process from analyzing big data could be formalized in software development.

With this improved formalization, I further attacked the problem of automating online A/B testing. I found the existing single-objective methods for this problem are quite limited in industrial settings in practice, so I proposed a **Multi-Objective Evolutionary Algorithm (MOEA)** for this problem [4]. However, this approach still only automates the parameter values in the system rather than more intelligent changes or features. Although going from automating local parameter optimization to non-trivial meaningful changes is challenging, recent advances in Generative AI (GAI) showed the remarkable capability of Large Language Models (LLM) in coding tasks. I plan to leverage LLM to further address this technical barrier to build data-driven decision systems for automatically delivering more intelligent changes in offline or online A/B testing.

Estimated effort: 3 months.
Outcome: a full research paper for a top-tier conference/journal.

Prototype system: After all of the four components in this research proposal are completed, I will have the **data-driven insights via A/B testing**, and the **conversational tool**, backed with **LLMs of stronger context-awareness, and communication skills** as foundations. I plan to put them together and build a prototype demo **for the "AI-Driven Development"** of a recommendation system, as a case study. I have extensive industrial experience in the recommendation system, which is a representative data-intensive, ML-enabled software application with heavy usage of A/B testing, data consumption, and ML in the industry. By building a practical prototype, I will further learn and identify any technical barriers toward this exciting goal.

Estimated effort: 3 months.

Additional resources: access to industrial applications is ideal but optional
Outcome: a prototype of an automated data-driven decision system for building a recommendation application. a demo paper to summarize the learning of building the prototype.

2. Understanding and Benchmarking the Communication Skills of Large Language Model for Code

Large language models (LLMs) have significantly improved the ability to perform tasks in the field of code generation. However, there is still a gap between LLMs being capable coders and being top-tier software engineers. In my recent work [1], based on the observation that top-level software engineers often ask clarifying questions to reduce ambiguity in both requirements and coding solutions, I argue that the same should be applied to LLMs for code generation tasks. I explored how to leverage better communication skills to achieve greater confidence in generated code. I proposed a **communication-centered process** that uses an LLM-generated communicator to identify issues with high ambiguity or low confidence in problem descriptions and generated code, then ask clarifying questions to obtain responses from users for refining the code.

Following this direction, I plan to create an **evaluation benchmark of communication skills** for LLM-based code generation. To make LLMs more like human, this is necessary to objectively quantify the capability of communication skills of LLM on code generation tasks and software engineering tasks. The existing evaluation work of LLMs for code generation mainly focus on solving algorithm problems without additional converstational input. I will target benchmark that reveals how effective is the communcation ability of the model. For examples, creating a dataset with blurred and noisy problem descriptions that hide some critical information is an interesting direction. In this setting, the model should ask the right questions rather than directly generate low-quality code.

Furthermore, another interesting angle is to study how to tune the model to switch between **under-communicating, effective-communcating and overcommunicating**. I envision that different AI programming agents in future will have various levels and styles of communication ability. This work can be seen as the first step toward improving communication skills of LLMs for code.

Estimated effort: 3 months.
Outcome: a full research paper for top-tier conference/journal.

Extension: Understanding and benchmarking other fundamental aspects of code language models. To extend further, besides the communication lens, I'm also interested in

applying other different lens with the goal of better understanding LLM for code. This includes (but not limited to) these aspects:

- documentation ability,
- probing,
- logical reasoning,
- program analysis,
- context-awareness,
- fine-tuning efficiency,
- cost

I view **explainability** as a top priority, because having a deep understanding will open up new ideas to further improve LLM for code, and open up new application of LLM to other software engineering tasks.

Estimated effort: 4-6 months.
Additional resources: GPU (for fine-tuning).
Outcome: a full research paper for top-tier conference/journal.

3. From Process Model to Conversational Tools for Building ML-Enabled Systems

As we are shifting from traditional software to ML-enabled systems to let computers automatically learn the parameters in programs, Machine learning (ML) components are being added to more and more critical and impactful software systems. This is an emerging field, called "**AI Engineering**", with a number of challenges on technical side, collaboration side and other sides. My recent work [2] focused on difficulties in using traditional software lifecycle models such as waterfall, spiral or agile model when building ML-enabled systems. I investigated the application of using systems engineering process in ML-enabled systems by interviewing with practitioners from multiple companies. I then developed **a set of propositions** and proposed **V4ML process model** for building products with ML components. I found that V4ML process model requires more efforts on documentation, system decomposition and V&V, but it **addressed the interdisciplinary collaboration challenges** and **additional complexity** introduced by ML components.

However, even with V4ML process model as guidance, developers and data scientists typically have very **limited bandwidth** in practice and are more likely to focus on top-priority tasks and push aside tasks suggested by V4ML. These tasks are of lower-emergency but essential in ensuring high-quality and reduced risk. I plan to develop a **conversational tool**, backed with LLM, to provide suggestions to developers guided by V4ML for elevated engineering quality and reduced risks when building ML-enabled systems. I have previous background and extensive experience in **building AI-based tools** such as **online sketch recognition** [7,8] and **offline sketch parsing** [6] for **efficient** flowchart creation. I will continue to bridge the

technical gaps when building tools, meanwhile focusing on the **efficiency, effectiveness, and usability of the tools** in completing the task.

Estimated effort: 3 months.
Outcome: a full research paper for a top-tier conference/journal, and a prototype of the tool.

4. Towards Context-Aware Code Language Models

Despite the great success of LLMs in coding assistants like GitHub Copilot, these models struggle to understand the context present in the artifacts and code repository, therefore generating inaccurate code. However, to realize my ultimate goal of creating autonomous data-driven decision systems, **knowing enough context in the artifacts**, including code repositories, internal/external documentation, and domain knowledge for the project, is critical.

With my research outcome in LLM listed above, I plan to **explore fine-tuning** (or even pre-training if needed) the LLM for **various code-related SE tasks** to enable **stronger context-awareness**.

Estimated effort: 4-6 months.
Additional resources: GPU (for fine-tuning).
Outcome: a full research paper for a top-tier conference/journal.

Impact of the Research Proposal: With the above four research components completed, the final output of this research proposal is the realization of the automated, data-driven decision systems for developing **data-intensive ML-enabled software applications**. The system will leverage **data-driven insights via A/B testing**, and the **conversational tool**, backed with **LLMs of better communication and context-awareness**. The decision systems will need the supervision of decision-makers to provide **guidance and feedback** when needed. The result further unlocks synergies between LLM (AI) and SE, and provides contributions to the exciting future of the transition from "**Data-Driven Development**" to "**AI-Driven Development**".

[1] Does Asking Clarifying Questions Increases Confidence in Generated Code? On the Communication Skills of Large Language Models
Wu, Jie JW
Arxiv. (Under Review)

[2] Application of Systems Engineering Process in ML-Enabled Systems
Wu, Jie JW
Arxiv. (Under Review)

[3] Towards Formalizing Data-Driven Decision-Making from Big Data: A Systematic Multi-Criteria Decision-Making Approach in Online Controlled Experiments
Wu, Jie JW
Ph.D. Dissertation

[4] A Multi-Objective Evolutionary Approach Towards Automated Online Controlled Experiments
Wu, Jie JW, Thomas A. Mazzuchi, and Shahram Sarkani
Journal of Systems and Software, 2023.

[5] Comparison of Multi-Criteria Decision-Making Methods for Online Controlled Experiments in a Launch Decision-Making Framework
Wu, Jie JW, Thomas A. Mazzuchi, and Shahram Sarkani
Information and Software Technology 155 (2023): 107115.

[6] Offline Sketch Parsing via Shapeness Estimation
Wu, Jie, Changhu Wang, Liqing Zhang, Yong Rui
IJCAI. Vol. 15. 2015.

[7] Sketch Recognition with Natural Correction and Editing
Wu, Jie, Changhu Wang, Liqing Zhang, Yong Rui
Proceedings of the AAAI Conference on Artificial Intelligence. Vol. 28. No. 1. 2014.

[8] Smartvisio: Interactive sketch recognition with natural correction and editing
Wu, Jie, Changhu Wang, Liqing Zhang, Yong Rui
Proceedings of the 22nd ACM international conference on Multimedia. 2014. (demo)

[9] Hidden technical debt in machine learning systems
Sculley, David, Gary Holt, Daniel Golovin, Eugene Davydov, Todd Phillips, Dietmar Ebner, Vinay Chaudhary, Michael Young, Jean-Francois Crespo, and Dan Dennison.
Advances in neural information processing systems 28 (2015).