Research Statement - Ziyang Li, University of Pennsylvania

The rapid integration of artificial intelligence into modern computing presents an unprecedented opportunity to advance programming languages and formal methods as foundational tools for building robust, efficient, and interpretable solutions. My research lies at the intersection of Programming Languages (PL) and Machine Learning (ML), with impactful applications spanning *cybersecurity*, *software engineering*, and the *life sciences*. I have developed Scallop, a neurosymbolic programming language, to bridge the gap between symbolic reasoning and neural learning. Scallop enables novel solutions to challenges in domains such as computer vision, natural language processing, program analysis, planning, clinical decision-making, and bioinformatics. My overarching goal is to design languages and frameworks that are *accessible*, *scalable*, and *applicable*, empowering researchers and practitioners to address increasingly complex problems with reliability, efficiency, and interpretability.

Specifically, I developed Scallop to address a fundamental challenge: combining the expressiveness of symbolic programming with the scalability of machine learning models in a single, cohesive framework. Scallop provides an intuitive language while ensuring effectiveness and usability for diverse applications. This work has resulted in a series of publications on core language design (PLDI'23 and AAAI'24), algorithms (NeurIPS'21 and NeurIPS'24), and applications (ACL'23, ICML'24 Spotlight, and two submissions to ICLR'25). Scallop has facilitated collaborations with researchers at Penn Medicine, CMU, and UC Berkeley. The research has been disseminated through summer school courses with hands-on labs (SSFT 2022 and SSNP 2024) as well as tutorials in premier conferences (LOG 2022 and PLDI 2023). It has been the focus of multiple funded grant proposals, including a dedicated NSF Medium grant titled *Scallop: A Neurosymbolic Programming Framework for Combining Logic with Deep Learning*.

Building on these foundations, my research agenda focuses on advancing the neurosymbolic infrastructure required for ML systems that are trustworthy, explainable, and efficient. I envision developing programming models and tools that seamlessly integrate with state-of-the-art AI technologies, such as large language models, to improve software quality and reliability. My work will continue to address challenges in scalability and accessibility, laying the groundwork for general-purpose neurosymbolic frameworks. I aim to democratize the use of advanced PL and ML techniques, making them accessible to a broader audience of researchers and practitioners.

1. Research Contributions

Classical algorithms and deep learning embody two prevalent paradigms of modern programming. Classical algorithms are well suited for exactly-defined tasks, such as sorting a list of numbers or finding a shortest path in a graph. Deep learning, on the other hand, is well suited for tasks that are not tractable or feasible to perform procedurally, such as detecting objects in an image or parsing natural language text. These tasks are typically specified using a set of input-output training data, and solving them involves learning the parameters of a deep neural network to fit the data using gradient-based methods.

While each paradigm is powerful in isolation, they are inherently complementary. Neurosymbolic programming is an emerging paradigm that integrates symbolic knowledge and reasoning with neural

architectures, offering the promise of improved efficiency, interpretability, and generalizability over neural or symbolic approaches alone. Despite substantial progress in individual neurosymbolic applications, there remains a critical gap: the absence of a unified programming language and compiler infrastructure to democratize the benefits of this paradigm and make it accessible to a broader audience.

To address this gap, my research contributions focus on advancing neurosymbolic programming by tackling three key challenges: **accessibility**, **scalability**, and **applicability**. These principles have driven the design and development of Scallop, a neurosymbolic programming language, resulting in a cohesive body of work spanning core language design, algorithmic innovation, and application-driven research.

Accessibility. Designing programming languages that are both expressive and easy to use is essential for broadening their adoption and impact. My work on Scallop focuses on creating a programming paradigm that lowers the barriers for users to harness the power of neurosymbolic methods. Scallop integrates a language, compiler, runtime, and supporting infrastructure, abstracting away the complexities of probabilistic and differentiable reasoning required for end-to-end integration within modern machine learning workflows. A core contribution, presented in PLDI'23, is Scallop's introduction of a **general provenance framework**, which allows developers to write programs as succinct logic rules while utilizing configurable reasoning backends. This design facilitates discrete, probabilistic, and differentiable reasoning, adapting to various neurosymbolic use cases. In subsequent work (AAAI'24), we expanded Scallop with foreign interfaces and an extensive plugin library, connecting it to modern foundation models. This enables structured reasoning with large language models and integrates Scallop with external databases such as CodeQL, allowing systematic analysis of real-world programs.

A well-designed programming interface has broadened the impact of neurosymbolic applications and established Scallop as a practical tool for researchers and practitioners. Scallop has been incorporated into **summer school courses**, including the Summer School on Formal Techniques (2022) and the Summer School of Neurosymbolic Programming (2024). These courses involve teaching Scallop's principles and practices, as well as hands-on labs to introduce students to this emerging paradigm. Additionally, Scallop has been featured in **tutorial sessions** at conferences such as LOG'22 and PLDI'23. In recognition of its impact, we were invited to author a book on Scallop for the *Foundations and Trends in Programming Languages* monograph series (2024). Scallop's development has also catalyzed inter-university collaborations and led to multiple funded grant proposals, including a $1.4M NSF-funded project titled *Scallop: A Neurosymbolic Programming Framework for Combining Logic with Deep Learning*.

Scalability. As a neurosymbolic programming language, Scallop must scale to handle real-life applications with complex use cases, often requiring the processing of vast amounts of in-the-wild data. A key challenge lies in the scalability of probabilistic and differentiable reasoning. Traditional exact probabilistic reasoning over discrete distributions requires enumerating all possible worlds, which can lead to exponential blow-ups. To address this, in NeurIPS'21, we introduced the **top-k proofs approximation algorithm**, which relaxes and generalizes exact probabilistic reasoning. This approach achieved orders-of-magnitude improvements over previous baselines like ProbLog and DeepProbLog. In PLDI'23, we extended this work by integrating top-k proofs into Scallop's provenance framework, allowing users to configure their reasoning backend with a choice between exact and various approximate reasoning algorithms. This unification enables users to balance reasoning granularity with scalability

based on their application needs. Further, in NeurIPS'24, we introduced ISED (Infer-Sample-Estimate-Descend), an algorithm that leverages only input-output samples to estimate gradients, improving both scalability and data efficiency.

In addition to algorithmic advancements, Scallop's scalability can be further enhanced by hardware acceleration. In a submission to PLDI'25, we developed Lobster, a **GPU-based runtime** for accelerating inference within Scallop. Lobster systematically translates key components of relational algebra into CUDA GPU kernels and implements a swappable runtime for maximum compatibility and efficiency. By fully supporting Scallop's provenance framework, Lobster seamlessly enables discrete, probabilistic, and differentiable modes of reasoning on GPU hardware. While Lobster has been tested across a variety of benchmarks—including neurosymbolic applications, probabilistic static analysis, and knowledge graph traversal—a standout use case is its application to RNA folding. This task, which involves folding long RNA sequences into secondary structures, traditionally requires significant computational resources. With Lobster's GPU runtime, Scallop achieved an **average speed-up of 146x** and a maximum speed-up of 500x compared to its CPU counterpart, reducing the runtime for inference on a complete RNA dataset from 13 hours to just 6 minutes.

These algorithmic and hardware accelerations not only address the inherent challenges of scaling neurosymbolic programming but also broaden Scallop's impact across a wide range of applications, positioning Scallop as a practical and powerful tool for real-world neurosymbolic computing.

Applicability. A programming language thrives when it demonstrates value through impactful use cases, which is why application-driven research is central to my work with Scallop. In fact, Scallop's design was initially inspired by my earlier research on AI-assisted software analysis (ICLR'20 Spotlight, S&P'21, and USENIX Security'24). While Scallop's capabilities in natural language reasoning and vision have been explored in foundational papers (PLDI'23 and AAAI'24), its applicability has expanded through numerous collaborative projects across a variety of domains. Key collaborations include:

- **Natural Language Reasoning**: In partnership with Prof. Eric Xing at CMU, I applied Scallop to long-dependency natural language reasoning tasks, fine-tuning language models and learning templated logic rules for downstream applications. This work achieved state-of-the-art performance on several benchmarks at the time and was published in ACL Findings'23.
- **Clinical Decision-Making**: In collaboration with Profs. Qi Long and Ravi Parikh at Penn Medicine, we integrated Scallop into workflows for cancer mortality prediction. Scallop serves as a backbone tabular database that bridges neural networks with interpretable explanation generators, resulting in impactful outcomes including an ICML'24 Spotlight paper.
- **Video Semantic Understanding**: Collaborating with Facebook AI Research and Prof. Ser-Nam Lim at UCF, we developed LASER, a weakly-supervised learning framework for video scene graph generation. This work, currently under review (ICLR'25), demonstrates Scallop's ability to facilitate structured reasoning in video understanding tasks.
- **Software Vulnerability Detection**: In collaboration with Prof. Saikat Dutta at Cornell University, I applied Scallop to whole-repository vulnerability detection. Scallop bridges CodeQL databases, which encode user programs, and large language models for API specification inference and trace-based contextual analysis. This work is currently under review (ICLR'25).

One of the most notable applications of Scallop is in bioinformatics, where I am collaborating with Prof. Li Shen at Penn Medicine on the problem of **RNA folding**. This project represents Scallop's first application in bioinformatics, with significant implications for downstream tasks such as RNA splicing and drug discovery. The task posed substantial scalability challenges due to the complexity of probabilistic reasoning over long RNA sequences. To address this, I designed compiler optimizations and language extensions tailored for bioinformatics workloads. Most notably, I utilized Lobster, the GPU runtime for Scallop, achieving orders-of-magnitude speed-ups that reduced runtime from hours to minutes. This collaboration not only showcases Scallop's applicability but also highlights the synergistic relationship between accessibility, scalability, and applicability: real-world applications drive the development of the underlying language, making it progressively more expressive and efficient.

2. Future Research Directions

Building on my work with Scallop, I aim to advance neurosymbolic programming as a core enabler for accessible, trustworthy, scalable, and generalizable AI systems. My future research will focus on three interconnected themes: expanding the frontiers of neurosymbolic programming frameworks, developing scalable and interpretable AI-powered tools for software quality, and exploring transformative applications across interdisciplinary domains.

Neurosymbolic Programming Frameworks. One immediate direction for my future research is to enhance the core capabilities of neurosymbolic programming frameworks. While Scallop has proven its potential in seamlessly combining symbolic reasoning with neural learning, there remain significant opportunities to improve expressiveness, scalability, and integration with emerging AI technologies. I plan to extend Scallop's expressiveness to support more complex reasoning paradigms, such as temporal reasoning in robotics and real-time systems, or causal reasoning for scientific discovery and decision-making tasks. Beyond Scallop, I aim to design novel language constructs that enable neurosymbolic frameworks to integrate seamlessly into large-scale AI agent systems and world models, bridging symbolic abstractions with the continuous nature of neural reasoning.

However, I believe the Datalog-based design of Scallop represents just one step forward. The future of neurosymbolic programming will likely involve the evolution of new programming paradigms that adapt to the demands of AI-driven computation. As such, I look forward to exploring entirely new programming languages that cater to the diverse needs of neurosymbolic systems, whether by introducing richer declarative semantics, incorporating probabilistic reasoning as a first-class citizen, or enabling adaptive reasoning for dynamic, data-driven environments.

AI-Assisted Software Development. The cycle of modern software development involves a complex pipeline of tasks, including programming, transpiling, debugging, vulnerability detection, and patching. The emergence of large language models presents unprecedented opportunities to enhance the scalability and usability of these processes, while also posing new reliability, safety, and interpretability challenges. By leveraging neurosymbolic techniques, I aim to design scalable, trustworthy tools that address challenges at every stage of software development.

Future iterations of Scallop could play a pivotal role in this vision by integrating with advanced program representations such as intermediate-level graphs, which provide a structured and scalable abstraction for analyzing and transforming programs. Additionally, Scallop's neurosymbolic framework could enable probabilistic reasoning to handle incomplete, noisy, or ambiguous software specifications, making it a powerful tool for tackling real-world challenges in modern software systems. Ultimately, I envision neurosymbolic methods as the foundation for a new generation of AI-assisted software engineering tools that democratize high-quality, secure, and efficient software development.

Interdisciplinary Applications. Expanding the scope of neurosymbolic programming to address interdisciplinary challenges is another key direction for my future research. Scallop has already shown its potential in fields such as natural language understanding, clinical decision-making, and bioinformatics. Building on these successes, I aim to explore how neurosymbolic frameworks can drive innovation in other domains by serving as a bridge between symbolic reasoning and data-intensive neural methods.

In bioinformatics, for instance, I plan to extend the work on RNA folding by addressing broader challenges such as RNA splicing, RNA modification, and drug discovering. By integrating neurosymbolic reasoning with probabilistic modeling, I envision frameworks that can handle incomplete or noisy biological data while offering interpretability and explainability. Another promising area is autonomous systems, where neurosymbolic programming can enhance temporal and causal reasoning required for robotics and high-level decision-making in dynamic environments. I plan to develop programming frameworks that support reasoning about actions and outcomes over time, enabling robots and autonomous agents to make informed decisions that align with high-level goals and constraints.

3. Conclusion

My research bridges the fields of programming languages and AI, focusing on advancing neurosymbolic programming to address complex, real-world challenges. Through the development of Scallop, I have contributed to making neurosymbolic programming more accessible, scalable, and applicable, enabling state-of-the-art results in diverse domains. I have not only demonstrated the potential of neurosymbolic methods but also inspired new programming language design, algorithmic innovation, and interdisciplinary applications.

Looking ahead, my research will continue to push the boundaries of neurosymbolic programming by addressing critical challenges in expressiveness, scalability, and integration with emerging AI technologies. I am particularly excited about exploring new programming paradigms that adapt to the evolving landscape of AI-driven computation, while extending the reach of neurosymbolic methods to domains such as autonomous systems, scientific computing, and life sciences. By pursuing these directions, I aim to make programming languages a cornerstone for the next generation of trustworthy, interpretable, and generalizable AI systems.

Beyond research, I am deeply committed to fostering collaboration and mentoring the next generation of computer scientists. I look forward to creating inclusive research environments, developing innovative teaching strategies, and inspiring students to contribute to cutting-edge interdisciplinary research. By combining foundational advancements with practical applications, my goal is to empower researchers and practitioners across disciplines to solve the pressing challenges of our time.