Workshop Proposal: Advances in Programming Languages and Neurosymbolic Systems (AIPLANS)

Anonymous Author(s)
Affiliation
Address
`email`

Abstract

Automatic differentiation libraries and frameworks have enabled much progress in gradient-based learning over the last decade. Recent domain-specific languages for automatic programming hold the promise of unleashing similar progress in e.g., probabilistic and classical reasoning. Concurrently, machines have made steady progress in representing and synthesizing programs. Other workshops have explored these themes separately, yet few have highlighted the synergies between automatic and synthetic programming, a situation we hope to remedy.

1 Introduction

Neural information processing systems have benefited tremendously from the availability of programming languages and frameworks for automatic differentiation. Similar domain-specific languages have shown progress automating inference in other logical disciplines, such as probabilistic and classical logic, proof nets, and related message passing schemes on tree- and graph-structured data.

Not only does machine learning itself benefit from languages for programmable inference, these systems can also be seen as a kind of low-level programming language in their own right, consisting of differentiable and stochastic primitives. While currently less interpretable, thanks to recent progress in statistical language modeling, these systems are increasingly capable of generating symbolic functions resembling procedures a human programmer might plausibly write in a high-level language.

Applying techniques from programmable inference to transform and generate programs, and adapting insights gained developing those same programs to drive innovation in higher-order AD and probabilistic programming is a virtuous cycle, with a growing stream of software and academic papers. We envision cooperation between automatic and synthetic programming will continue to increase as researchers become more accustomed to outsourcing low-level reasoning tasks to these systems.

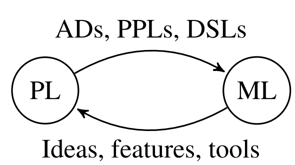

Figure 1: The ML/PL virtuous cycle.

Many ideas are being reinvented and rediscovered in this process. AD itself was invented a half dozen times over the last century and research continues to reveal unexpected connections to implicit differentiation, bilevel optimization, optimal control, stochastic processes and differential equations.

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021).

Semiring programming has existed in various forms for many decades and shares deep connections to reinforcement learning, structured inference and probabilistic programming. Much work remains.

Likewise, many recently-transplanted ideas in machine learning are catechism in the programming language literature. For example, functional and type-safe programming are lingua franca in PL circles but relatively new to Python, the primary language used in machine learning. The duality between code and data is well-known in PL under the aegis of homoiconicity. PL theory has thought deeply about catgorical semantics, concurrency, process calculi, linear logic, privacy and other deeply useful concepts which remain, to this day, mostly unfamiliar in the machine learning community.

Similarly, the programming language community too, has its blind sppots. PLs have long wrestled with the distinction between intensional and extensional representation, a distinction which the statistical learning community has long since reconciled under the umbrella of model-based learning and approximation theory. PL could take a page from structured inference and propagation algorithms as a medium for distributed computation. We believe many other such examples await discovery.

Other areas where the interaction could be fruitful are tools for equivalence, proof search and metrics. A deeper understanding of programming language semantics are largely missing from neural program synthesis discussions. The connection between various forms of message passing in concurrent systems and neural science merits further investigation. New language models could enable more effective tools for natural language and assistive programming. While some of these topics remain greenfield research topics, many connections are known, but yet-to-be-translated textbook knowledge.

As outlined above, we believe that recent advances in statistical learning and programming languages have been largely siloed, but these two communities have many ideas yet to share. In exchange, we believe a great deal of progress can be achived, in particular, between automatic and synthetic programming. A joint workshop such as the one put forward in this proposal could help to facilitate yet-unrealized research connections among neighboring fields. Our workshop is designed to be as inclusive as possible towards researchers of various backgrounds working on programming languages and neurosymbolic systems. For illustration, we include the following non-exhaustive list of topics:

- Differentiable programming / algorithmic differentiation
- Probabilistic programming / statistical inference
- Declarative programming / constraint programming
- Dynamic programming / reinforcement learning
- Functional programming / λ-calculus
- Array programming / linear algebra
- Semiring programming / message passing
- Logic programming / Relational programming
- Meta-programming / meta-learning
- Computer aided reasoning / automatic theorem proving
- Domain-specific languages and compilers
- Inductive programming / programming by example
- Genetic programming / evolutionary algorithms
- Differential privacy / algorithmic fairness

We would like to particularly encourage developers of languages, frameworks and libraries to submit their ongoing work for evaluation. Those who traditionally publish in venues such as SIGPLAN and SIGSOFT are also encouraged to consider submitting work that may be relevant to machine learning community. Details regarding evaluation criteria, deadlines and workshop logistics will be made available in a timely manner, pending acceptance. Further information, including examples of relevant literature can be found at `https://aiplans.github.io`.