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 interplay 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 abstractions for automatic differentiation. Similar domain-specific languages have begun to automate 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 tools and languages for programmable inference, learning can also be seen as a programming language of sorts that humans program indirectly. This is increasingly capable of synthesizing human-readable procedures. Examples of synthetic functions of this sort are starting to emerge, thanks to recent progress in statistical language modeling, resembling procedures a human programmer might plausibly write by hand.

Using techniques from programmable inference to transform and generate programs, and adapting insights gained developing those programs to drive innovation in 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 unlock deeper insights as researchers become more accustomed to outsourcing low-level reasoning tasks to these systems.

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, optimal control, stochastic processes and differential equations. 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 recent topics in machine learning have been studied 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. Other PLs have thought deeply about higher-order functions, currying, partial application, and denotational and operational semantics, which enables routines to interoperate smoothly and run reliably.

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

Similarly, the programming language community has wrestled with the distinction between intensional and extensional representation, a distinction which the statistical learning community has long since come to terms with 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. New language models could enable natural language and assistive programming.

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 to exchange. In particular, the connection between automatic and synthetic programming deserves further attention. A joint workshop such as the one put forward in this proposal could help to facilitate yet-unrealized research connections. Our workshop is designed to be as inclusive as possible. For illustration, we include the following non-exhaustive list of topics:

- Differentiable programming / automatic 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
- Metaprogramming / reflection
- Logic programming / proof search
- Domain-specific languages

We encourage developers of languages, frameworks and libraries to submit their ongoing work for evaluation. Further details regarding evaluation criteria, deadlines and workshop logistics will be made available promptly, pending acceptance.