AI-DSL Technical Report (May to Septembre 2022)

Nil Geisweiller, Samuel Roberti

October 18, 2022

Abstract

Contents

Chapter 1

Introduction

1.1 Setting the Scene

In the previous iteration we explored using Dependent Types to express formal specifications of AI services, with the ultimate goal of building a language for easily writing those specifications, the AI-DSL itself, as well as services to automatically connect AI services together, the AI-DSL Registry [1].

Back then we experimented with trivial AI services, computing simple arithmetic, described by trivial properties, such as the parity of their inputs/outputs. We were able to demonstrate that Idris, our DTL of choice, could be used to verify the correctness of such AI service assemblages. The approach seemed promising, but to really put the idea to the test we had to make progress on two fronts:

1. Replace trivial AI services by actual AI algorithms.
2. Explore program synthesis, as it became clear that it was at the heart of this endeavor. First, for building the AI service assemblages themselves. Second, for achieving fuzzy matching, that is when AI services almost fit together but not quite yet. And third, for completing assemblages when some AI services are outright missing.

That is what we have done during that iteration.

1.2 Work Accomplished

First we have implemented three AI algorithms in Idris:

1. Gradient descent
2. Linear regression
3. Logistic regression

These algorithms were chosen because they are relatively simple, yet extensively use in real world applications, as well as tightly related to each other. Linear regression can be framed as a gradient descent problem, and logistic regression can be framed both as gradient descent and linear regression problems, thus constituting an excellent case study for the AI-DSL. Alongside these implementations, a descending property was formulated and formally proved for each algorithm.

Finally, we have explored ways to perform program synthesis of dependently typed programs. While we have only achieved partial success as far as program synthesis is concerned, we were able to demonstrate its feasibility within the Idris ecosystem. It was clear from the start anyway that to be done well and fully, program synthesis essentially requires achieving AGI. Indeed, it is one of these AI-complete problems. That is any problem can be framed as a program synthesis problem and vice versa. The idea being that such functionality can be progressively grown, deferred less and less to human intervention, as the network and the AI-DSL evolve.

1.3 Related Work

Here's a list of projects and publications we have discovered along the way that relate to the work done during that iteration. TODO

Chapter 2

Implementation and Verification of AI Algorithms

Chapter 3

Program Synthesis

3.1 Language Framework

3.2 Idris Elaboration

3.3 Idris Proof Search

Idris proof search, contrary to what its name may suggest, can be used for program synthesis in general, not just proof search – which is no surprise to those familiar with the Curry-Howard correspondence.

Chapter 4

Conclusion

Appendix A

Glossary

- **AI service assemblage**: collection of AI services interacting together to fulfill a given function. Example of such AI service assemblage would be the Nunet Fake News Warning system.
- **Dependent Types**: types depending on values. Instead of being limited to constants such as `Integer` or `String`, dependent types are essentially functions that take values and return types. A dependent type is usually expressed as a term containing free variables. An example of dependent type is `Vect n a`, representing the class of vectors containing `n` elements of type `a`.
- **Dependently Typed Language**: functional programming language using dependent types. Examples of such languages are Idris, AGDA and Coq.
- **DTL**: Shorthand for Dependently Typed Language.

Bibliography

[1] Nil Geisweiller, Kabir Veitas, E.S.A.S.R.M.I.B.G.: AI-DSL Technical Report (February to May 2021) (2021), `https://github.com/singnet/ai-dsl/blob/master/doc/technical-reports/2021-May/ai-dsl-techrep-2021-05_may.pdf`