Linear logic and recurrent neural networks

Daniel Murfet

November 5, 2016

1 Introduction

One of the oldest problems in artificial intelligence is *program induction*: given a set of input-output pairs (say of binary sequences), the problem is to produce a program which generalises the given pairs. There is a large literature on this problem [**?**, **?**] but it has received renewed attention in the past few years due to the advances in deep learning [**?**, **?**, **?**].

Arguably this problem is hard because learning methods are generally continuous, whereas programs (realised as say Turing machines, or λ-terms, or C programs, etc) are basically discrete. In this paper we initiate a novel two-step approach to the problem. In this first step, we couple a standard recurrent neural network to "differentiable programmatic components" which are vectors in a vector space which is the denotation of a type in linear logic. Once the optimisation algorithm has found a good vector representative of programs in this space, we apply the second step, which uses proof search to find a proof in linear logic whose denotation is observationally equivalent to the vector program.

References