Biologically Inspired Cognitive Architectures xxx (xxxx) xxx–xxx

Contents lists available at ScienceDirect

Biologically Inspired Cognitive Architectures

journal homepage: www.elsevier.com/locate/bica

Research article

Learning of human-like algebraic reasoning using deep feedforward neural networks

Cheng-Hao Cai[a], Yanyan Xu[b,*], Dengfeng Ke[c], Kaile Su[d]

[a] Department of Computer Science, The University of Auckland, 38 Princes Street, Auckland 1142, New Zealand
[b] School of Information Science and Technology, Beijing Forestry University, 35 Qing-Hua East Road, Beijing 100083, China
[c] National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, 95 Zhong-Guan-Cun East Road, Beijing 100190, China
[d] Institute for Integrated and Intelligent Systems, Griffith University, 170 Kessels Road, Nathan, QLD 4111, Australia

ARTICLE INFO

2010 MSC:
00-01
99-00

Keywords:
Reasoning-based learning
Deep learning
Algebraic reasoning
Neural network reasoning

ABSTRACT

Human-like rewriting, which is an algebraic reasoning system imitating human intelligence of problem solving, is proposed in this work. In order to imitate both learning and reasoning aspects of human cognition, a deep feedforward neural network learns from algebraic reasoning examples produced by humans and then uses learnt experiences to guide other reasoning processes. This work shows that the neural network can learn human's behaviours of solving mathematical problems, and it can indicate suitable directions of reasoning, so that intelligent and heuristic reasoning can be performed. Moreover, human-like rewriting bridges the gap between symbolic reasoning and biologically inspired machine learning. To enable the neural network to recognise patterns of symbolic expressions with non-deterministic sizes, the expressions are reduced to partial tree representations and then vectorised as numeric features. Further, the centralisation method, symbolic association vectors and rule application records are used to improve the vectorised features. With these approaches, human-like rewriting shows satisfactory performance on the tasks of solving linear equations and computing derivations and indefinite integrals.

Introduction

Reasoning and learning are two main aspects of human cognition. Human reasoning can be considered as a process of using previously acquired knowledge to solve problems, and this process can be simulated using heuristic search (Duris, 2018). Human learning, on the other hand, can be considered as a process of acquiring knowledge from the real world, and this process can be simulated using artificial neural networks which is a kind of biologically inspired machine learning models (Rasheed, Amin, Sultana, & Bhatti, 2017). In order to imitate human cognition, machines are expected to imitate both reasoning and learning abilities. Although the reasoning ability could be imitated via symbolic reasoning techniques, and the learning ability could be imitated via machine learning techniques, it was still challenging to integrate symbolic reasoning and machine learning in effective ways (Garcez et al., 2015). In the field of symbolic reasoning, much work has been done on using formal methods to model reliable reasoning processes (Chang & Lee, 1973). For instance, algebraic reasoning could be modelled using first-order predicate logics or even higher-order logics, and these logics were usually designed by experienced experts (Bundy & Welham, 1981; Nipkow, Paulson, & Wenzel, 2002). In the field of machine learning, deep architectures were used to learn speech features and image features from data (Lecun, Bengio, & Hinton, 2015; Mohamed, Dahl, & Hinton, 2012; Sun, Liang, Wang, & Tang, 2015). However, symbolic reasoning and machine learning seemed to be two independent processes. The connection between the two processes was relatively weak. As a result, it was difficult for a machine to imitate simultaneously both reasoning and learning abilities. To bridge the gap between symbolic reasoning and machine learning, this research explores the possibility of using deep architectures to learn logics of algebraic reasoning. In this research, deep neural networks are trained to solve mathematical problems, such as finding the solution of an equation and calculating the differential or integral of an expression.

Rewriting is frequently used in the field of algebraic reasoning. The core concept of rewriting is to simplify a reasoning process by replacing expressions with equivalent ones (Bundy, 1983). Usually, rewriting is based on a tree-manipulating system, as many algebraic expressions can be represented by using tree structures, and the manipulation of symbols in the expressions is equivalent to the manipulation of nodes, leaves and sub-trees on the trees (Rosen, 1973). To manipulate symbols,

* Corresponding author.
E-mail addresses: ccai606@aucklanduni.ac.nz (C.-H. Cai), xuyanyan@bjfu.edu.cn (Y. Xu), dengfeng.ke@nlpr.ia.ac.cn (D. Ke), k.su@griffith.edu.au (K. Su).

https://doi.org/10.1016/j.bica.2018.07.004
Received 14 April 2018; Received in revised form 30 June 2018; Accepted 5 July 2018

a rewriting system usually uses one way matching, which is a restricted application of unification, to find a desired pattern from an expression and then replaces the pattern with another equivalent pattern (Bundy, 1983). In order to reduce the search space, rewriting systems are expected to satisfy the properties of termination and local confluence (i.e. the Church-Rosser property). In other words, the termination property requires that any rewriting process can eventually stop, and the local confluence property requires that any expression can be rewritten as one and only one simplest form. Totally, the expression should be rewritten as the simplest form within finite steps (Huet, 1980; Rosen, 1973). The formal definitions of rewriting, termination and local confluence are provided by Definitions 1–3 in Section "Standard rewriting". In order to make a system be terminating and locally confluent, the design of the system may start from small sub-systems, because proving termination and local confluence of a smaller system is usually easier than proving those of a larger system (Bundy & Welham, 1981). After multiple sub-systems have been designed, they can be combined into a whole system. The whole system is terminating and locally confluent, because the direct sum of two Church-Rosser systems holds the same property (Toyama, 1987). Some previous work has focused on the Church-Rosser property of rewriting systems. For example, the Knuth-Bendix completion algorithm was used to solve the problem of local confluence (Knuth and Bendix, 1983), and Huet (1981) provided a proof of correctness for this algorithm. Additionally, dependency pairs and semantic labelling were used to solve the problem of termination (Arts & Giesl, 2000; Zantema, 1995).

Deep architectures have been used in many fields of artificial intelligence, including speech recognition (Mohamed et al., 2012), human face recognition (Sun et al., 2015), natural language understanding (Sarikaya, Hinton, & Deoras, 2014), reinforcement learning for playing video games (Mnih et al., 2015) and Monte Carlo tree search for playing Go (Silver et al., 2016). Recently, the use deep architectures to simulate reasoning behaviours has become an emerging topic. For instance, Irving et al. (2016) have proposed DeepMath which uses various deep neural networks to guide premise selection of automated theorem proving. Moreover, Serafini and Garcez (2016) have proposed logic tensor networks to combine deep learning with logical reasoning. Further, Garnelo, Arulkumaran, and Shanahan (2016) have used reinforcement learning to perform the interaction between symbolic reasoning and neural learning. Additionally, Cai, Ke, Xu, and Su (2017) have used multi-layer perceptrons to manipulate variables in logical expressions and discover knowledge from the logical expressions.

In this research, deep architectures are used to guide algebraic rewriting processes. This technique is called human-like rewriting, as it is adapted from standard rewriting and can imitate human's behaviours of using rewrite rules after learning from algebraic reasoning schemes. Contributions of this work are listed as follows.

- Human-like rewriting, which is the combination of standard rewriting and artificial neural networks, has been proposed in this work.
- This work has demonstrated that deep feedforward neural networks are able to learn human's behaviours of reasoning and use learnt experiences to guide other reasoning processes.
- This work proposed vectorised representations of symbolic expressions, which bridges the gap between formalised reasoning and biologically inspired machine learning.

The rest of this article is organised as follows. Section "Standard rewriting" reviews standard rewriting. Section "Human-like rewriting" introduces human-like rewriting which is the core method of this work. Section "Methods for system improvement" introduces three methods for system improvement. Section "Experiments" evaluates the proposed human-like rewriting and the three improvement methods. Section "Conclusion" concludes this work.

Standard rewriting

Rewriting is an inference technique for replacing expressions or subexpressions with equivalent ones. The definition of rewriting is as follows (Bundy, 1983).

Definition 1 (*Rewriting*). Rewriting requires a source expression s and a set of rewrite rules τ. Let $l \Rightarrow r$ denote a rewrite rule in τ, t a subexpression of s, and θ the most general unifier of one way matching from l and t. A single rewriting step of inference can be formed as:

$$\frac{s(t) \quad (l \Rightarrow r) \in \tau \quad l[\theta] \equiv t}{s(r[\theta])} \tag{1}$$

Standard rewriting is to repeat the above step until no rule can be applied to the expression further.

In Eq. (1), θ is only applied to l, but not to t. The reason is that one way matching, which is a restricted application of unification, requires that all substitutions in a unifier are only applied to the left-hand side of a unification pair. Below is an example of rewriting.[1] Given two rules of the Peano axioms[2]:

$$x + 0 \Rightarrow x \tag{2}$$

$$x + S(y) \Rightarrow S(x + y) \tag{3}$$

$S(0) + S(S(0))$ can be rewritten via:

$$\begin{aligned} & \underbrace{S(0) + S(S(0))}_{by\ Eq.(3)} \\ \Rightarrow\ & S(\underbrace{S(0) + S(0)}_{by\ Eq.(3)}) \\ \Rightarrow\ & S(S(\underbrace{S(0) + 0}_{by\ Eq.(2)})) \\ \Rightarrow\ & S(S(S(0))) \end{aligned} \tag{4}$$

In Eq. (1), the set of rewrite rules τ is expected to satisfy the properties of termination. The termination property requires that any rewriting process can eventually stop. Formally, the termination property is defined as follows (Huet, 1980).

Definition 2 (*Termination of Rewriting*). A set of rewrite rules τ is terminating if any expression s can be rewritten as a normal form t such that no rewrite rule can be applied to t.

For instance, the chain rule in calculus $\frac{D(f)}{D(x)} \Rightarrow \frac{D(f)}{D(u)} \cdot \frac{D(u)}{D(x)}$ can result in non-termination:

$$\begin{aligned} & \frac{D(Sin(X))}{D(X)} \\ \Rightarrow\ & \frac{D(Sin(X))}{D(u_1)} \cdot \frac{D(u_1)}{D(X)} \\ \Rightarrow\ & \frac{D(Sin(X))}{D(u_2)} \cdot \frac{D(u_2)}{D(u_1)} \cdot \frac{D(u_1)}{D(X)} \\ \Rightarrow\ & \frac{D(Sin(X))}{D(u_3)} \cdot \frac{D(u_3)}{D(u_2)} \cdot \frac{D(u_2)}{D(u_1)} \cdot \frac{D(u_1)}{D(X)} \\ \Rightarrow\ & \cdots \end{aligned} \tag{5}$$

The above process means that the chain rule may not be used in standard rewriting, though it is an important rule. Similarly, the commutativity rule $x \circ y \Rightarrow y \circ x$, where $\circ$ is an addition, a multiplication, a logical conjunction, a logical disjunction or another binary operation satisfying commutativity, may not be used in standard rewriting.

Moreover, in Eq. (1), the set of rewrite rules τ is expected to satisfy the properties of local confluence. Local confluence requires that all branches of rewriting can lead to the same result. It can be defined as

[1] We use the mathematical convention that a word is a constant if its first letter is in upper case, and it is a variable if its first letter is in lower case.

[2] More detailed discussions about the Peano axioms can be found in (Pillay, 1981).