Available online at www.sciencedirect.com

ScienceDirect

Engineering Applications of Artificial Intelligence 19 (2006) 741–752

Engineering Applications of
ARTIFICIAL INTELLIGENCE

www.elsevier.com/locate/engappai

Imitation learning with spiking neural networks and real-world devices

Harald Burgsteiner*

Department of Information Engineering, InfoMed/Health Care Engineering, Graz University of Applied Sciences, Eggenberger Allee 11, A-8020 Graz, Austria

Received 18 May 2006; accepted 21 May 2006
Available online 18 July 2006

Abstract

This article is about a new approach in robotic learning systems. It provides a method to use a real-world device that operates in real-time, controlled through a simulated recurrent spiking neural network for robotic experiments. A randomly generated network is used as the main computational unit. Only the weights of the output units of this network are changed during training. It will be shown, that this simple type of a biological realistic spiking neural network—also known as a neural microcircuit—is able to imitate robot controllers like that incorporated in Braitenberg vehicles. A more non-linear type controller is imitated in a further experiment. In a different series of experiments that involve temporal memory reported in Burgsteiner et al. [2005. In: Proceedings of the 18th International Conference IEA/AIE. Lecture Notes in Artificial Intelligence. Springer, Berlin, pp. 121–130.] this approach also provided a basis for a movement prediction task. The results suggest that a neural microcircuit with a simple learning rule can be used as a sustainable robot controller for experiments in computational motor control.

Keywords: Robotics; Learning; Spiking neural networks

1. Introduction

Living and moving creatures most perfectly exhibit abilities that are the focus of many research areas. Some of them will be discussed in parts in this article: information processing of real-world sensors in real-time, learning of complex tasks on a variety of time scales and controlling actuators to move around in the habitat. The idea of creating artificial creatures that can move autonomously and exhibit a certain learning behaviour is not new. Many successful approaches have been found to accomplish some tasks that involve the areas mentioned above. These include e.g. the use of

- reinforcement learning techniques that reward or punish a subject for good or bad actions, respectively,
- genetic algorithms that "evolve" robots over many generations,
- echo-state networks that incorporate recurrent artificial neural networks (ANNs) and finally also,
- approaches based on biologically realistic neural networks (i.e. "spiking neural networks", SNNs).

Often combinations of these techniques are used. Some of them will be briefly discussed in this introduction. One of the most fascinating approach seems to be the last one, since advances in the research would include to enable us to understand parts of the biological processes that occur in everyone of us. Furthermore, SNNs are proven to be computationally more powerful than conventional ANNs (Maass, 1997). This can be interpreted in a way that a neural network consisting of spiking neurons needs fewer nodes to solve a problem than an ANN. Hence, e.g. an implementation of a robot controller in silicon would need less space and also less electrical power. SNNs also exhibit other interesting features that recommend them for use in real-world robot control, like noise robustness and the need for only simple communication mechanisms, as spikes can be modelled as binary events that occur on certain time points. In contrast to that, ANNs in most cases require

*Corresponding author. Tel.: +43 316 5453 6515; fax: +43 316 5453 96515.
E-mail address: harald.burgsteiner@fh-joanneum.at.
URL: http://www.igi.tugraz.at/harry/.

0952-1976/$ - see front matter © 2006 Elsevier Ltd. All rights reserved.
doi:10.1016/j.engappai.2006.05.007

high precision analog values that have to be communicated between single nodes of the network. Hence, controllers based on SNNs would be ideal candidates for autonomous robots. Still, known approaches for learning architectures and algorithms for SNNs rely mostly on time-consuming genetic algorithms. This will we explained in more detail in Section 1.2. Though an extensive search in literature has been done, no other usable implementations (apart from genetic algorithms) of complex spike-based controllers for robots that operate in real-time and employ learning mechanisms have been found. Hence, this article mainly focuses on two areas: (a) learning in biologically realistic neural networks and (b) robotics using neural network controllers.

1.1. Robotics based on ANNs

The term ANN commonly refers to a net of neurons that do not try to be biologically realistic, in that sense that they do not imitate behaviours of biological neurons (e.g. action potential generation, post-synaptic potential shaping, follow refractory periods, model dynamical synapses, etc.). Instead, one tries to model neurons that exhibit a computationally similar behaviour as a single biological neuron or even as a pool of biological neurons. Typical examples are multi-layer perceptrons, radial basis function networks or self-organizing maps.

The common view of a neural network is that of a set of neurons plus a set of weighted connections (synapses in the biological context) between the neurons. Each neuron comes with a transfer function computing an output from its set of inputs. In multi-layer networks these outputs can again be used as an input to the next layer of neurons, weighted by the relevant synaptic "strength". Feed-forward networks have only connections starting from external input nodes, possibly via one or more intermediate hidden node processing layers, to output nodes. Recurrent networks may have connections feeding back to earlier layers or may have lateral connections (i.e. to neighbouring neurons on the same layer). See Fig. 1 for a comparison of the direction of computation between a feed-forward and a recurrent neural network. With this recurrency, activity can be retained by the network over time. This provides a sort of memory within the network, enabling them to compute functions that are more complex than just simple reactive input–output mappings. This is a very important feature for networks that will be used for generating adaptive behaviour in robotics, because in most cases the current behaviour of the robot is not solely a function of the current sensory input, but a function of the current and previous sensory inputs and also of the current and previous internal network states. This allows a system to incorporate a much richer range of dynamical behaviours. Many approaches have been elaborated on recurrent ANNs. Some of them are

- dynamic recurrent neural networks (with backpropagation through time, see Pearlmutter, 1995),
- radial basis function networks (when one views lateral connections also as a recurrency) from Bishop (1995),
- Elman networks (recurrency through a feedback from the hidden layer onto itself, see Elman, 1990),
- self-organizing maps (Kohonen, 2001),
- Hopfield nets (Hopfield, 1982) and
- the "echo-state" approach from Jäger (2001) which will be covered in more detail in Section 1.3.

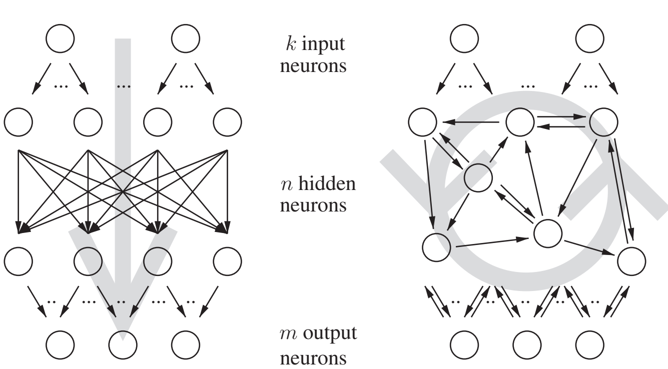

Fig. 1. Comparison of the architecture of a feed-forward (left-hand side) with a recurrent artificial neural network (right-hand side); the grey arrows sketch the direction of computation.

In case of autonomous agents it is rather difficult to employ strictly supervised learning algorithms for recurrent ANNs such as backpropagation, Boltzmann machines or learning vector quantization (LVQ), because the correct output is not always available or computable. It is also very difficult to set the weights of a recurrent ANN directly for a given non-trivial task. Hence, other learning techniques have to be developed for ANN, that could simplify the learning process of complex tasks for autonomous robots. Two approaches—genetic algorithms and echo-state networks—will be reviewed in the next two subsections.

1.2. Genetic algorithms

Genetic algorithms (Goldberg, 1989) are one approach that has been used widely to employ ANNs in robot control. The idea behind genetic algorithms in robotics is that one does not have to specify a controller or a target function directly (which would be required by e.g. a learning algorithm like backprop). Instead, one tries to let autonomous agents evolve "automatically" over many generations. In most cases, simple genetic algorithms are being applied. Floreano and Mondada (1996) give a good description of experiments with genetic algorithms in robotics (with a set-up similar to the one used in this article). In principle different parameters of the system that should evolve are coded in e.g. bitstring (as its "chromosome"). An initial population with variations in these parameters is formed. Each individual has a given "life-time" in which it can perform according to the settings in its chromosome, after which the performance is evaluated