A TOOL FOR FAST DEVELOPMENT OF MODULAR AND HIERARCHIC NEURAL NETWORK-BASED SYSTEMS

Francisco Reinaldo
LIACC, Faculty of Engineering, University of Porto, Portugal
E-mail: reifeup@fe.up.pt

Mauro Roisenberg
L3C, Department of Computer Science, Federal University of Santa Catarina, Brazil

Jorge Muniz Barreto
L3C, Department of Computer Science, Federal University of Santa Catarina, Brazil

Rui Camacho
LIACC, Faculty of Engineering, University of Porto, Portugal

Luis Paulo Reis
LIACC, Faculty of Engineering, University of Porto, Portugal

KEYWORDS: Agent, Behaviour, Architecture, PyramidNet

ABSTRACT

This paper presents the PyramidNet Tool as a fast and easy way to develop Modular and Hierarchic Neural Network-based Systems. This tool facilitates the fast emergence of autonomous behaviours in agents because it uses a hierarchic and modular control methodology of heterogeneous learning modules: the pyramid. Using the graphical resources of PyramidNet the user is able to specify a behaviour system even having little understanding of artificial neural networks. Experimental tests have shown that a very significant speedup is attained in the development of modular and hierarchic neural network-based systems when using this tool.

1. INTRODUCTION

Cognitive processes are necessary to reach autonomous behaviour. A cognitive process is a decision-making route of beliefs or desires that triggers actions for emerging behaviours. In order to be autonomous, an agent has to include processes like representation, manipulation, combination and modification of behaviours. Other processes like perception, learning, deduction and planning are important as well (Wooldridge and Jennings 1994).

This paper presents the PyramidNet Tool (Reinaldo 2003) as a fast and easy way to develop a modular and hierarchic neural network-based system. Inspirited in PyramidNet Architecture (Roisenberg, Barreto et al. 2004), the user can interact with the Tool and create different projects using Artificial Neural Networks (ANN).

The paper is divided in six sections. Section 2 introduces the PyramidNet Architecture and features. Section 3 introduces the PyramidNet Framework and sub-frameworks. Section 4 presents the PyramidNet Tool and features. Section 5 presents an experiment and discusses the results. In Section 6, we draw some conclusions.

2. PYRAMIDNET ARCHITECTURE

The PyramidNet Architecture (Roisenberg, Barreto et al. 2004) uses a modular and hierarchical approach of ANNs to emerge reasoning to produce behaviours in agents, as shown in Figure 1. The use of ANNs represents many advantages over other proposed control architectures because it supports high noise immunity, fault tolerance and programming by examples.

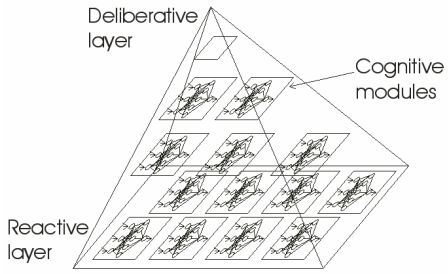

Figure 1: PyramidNet Architecture

The structure of the pyramid is arranged to provide a fast emergence of highly flexible behaviours. The hierarchic structure is used to separate levels of complexity into incremental functionality. It is composed by multiple levels of function. These levels/layers of function represent subsequent clusters of ANN arranged in a hierarchical way, allowing increasingly behaviours like a cortex (neocortex) (Kolb and Whishaw 2005). The base of the pyramid runs reactions (real-time activities), exploring the straightforward performance in the effector level, and the top of the pyramid has responsibility for producing of inner representations (cognitive reasoning), more elaborated behaviours for controlling the reactive levels. Therefore, layers and learning modules communicate through interconnections.

3. PYRAMIDNET FRAMEWORK

PyramidNet Framework (Reinaldo 2003) is an open-source C++ platform that offers a robust group of reusable objects/classes for developing future flexible and extensible ANN tools. The main objective is to develop and standardize a fast reliable procedure for tools that support

the layered organization of learning modules. The framework enables the construction of a tool through union/extension of classes and communication between objects to create solutions to similar problems (Wirfs-Brock and Johson 1990).

The framework has three main sub frameworks, which are the Graphic User Interface Framework (GUIF), the Artificial Neural Networks Framework (ANNF) and the Automatic Open-Source Code Generator Framework (ACF). GUIF is a set of main classes for handling and modelling learning processes, using graphic elements that will interact with users. It makes some items available: desktop constructor, sensor, actuator, line links, skins of ANN, scheduler of ANN training, menus, dialog box and others. ANNF offers classes of learning modules. Each ANN class is composed of algorithms, layers, neurons and its respective particularities still to be worked. In addition, it is implemented to be extensible for receiving more classes. ACF offers a set of classes able to interpret the drawn project. Besides, it can be extended to generate an automatic open-source code of a program.

4. PYRAMIDNET TOOL

PyramidNet Tool (Reinaldo 2003) is a utility that helps students/users to develop Modular and Hierarchic Neural Network-Based Systems. The Tool addresses Artificial Intelligence, Psychology and Robotics areas. The Tool machinery was built over some classes of PyramidNet Framework because they maintain standard and collaborative tasks.

PyramidNet Tool provides three ANN learning modules that are Feedforward networks (Fine 1999), Recurrent networks (Mandic, Chambers et al. 2001) and SOM – Kohonen's maps (Kohonen 2001). These modules can reach the behaviour levels in order: Stereotyped (reflexive and taxies), Reactive and Deliberative. Anything less is like developing behaviours to an agent but letting fine movements and elaborated decisions without being used. The Tool permits users to make several tests using one isolated type or several heterogeneous ANN in communication. The features of the Tool include many aspects. The first of these is the fact of enabling users to design and implement new behaviours so that agents run in dynamic environment, using simple objects in a graphic environment as well as minimum knowledge of programming language. These functionalities enable the simplest, lightest and fastest way to produce behaviour projects to agents. Other feature is the ability to manipulate models because humans are good at using diagrams, and it is not necessary to adopt new practices to use the tool. There are parameters to be set, and the level of description is clean. The most important feature of the tool is the ability to train several learning modules using information from sensors and/or other learning modules making an array of learning modules for the present tool's process. Taking specific input data and producing specific output data in a flows that define the sequential interrelation between activities and can be specified with either linear or branched connections. Another useful feature is the ability to produce a clean and ready-to-use ANSI C open-source for information fusion, planning and coordination with a few mouse clicks. By using a high-performance interpretation algorithm, a functional and compact core is created from drawn project. This allows the .cpp code to be changed easily. Other features are things like customizable interfaces, a system allowing integration of other projects that may assist customer support in helping the consumers. All these features of the PyramidNet Tool has been used as curricular component on the master degree course of computer science in the Federal University of Santa Catarina (UFSC) – Brazil -, and Laboratory of Connectionism and Cognitive Computing (L3C) – Brazil.

The open-source PyramidNet Tool can be fully downloaded at (Reinaldo). More details about the development of the Tool can be seen in (Reinaldo; Reinaldo 2003). The next section presents an experiment which was created by using PyramidNet Tool.

5. EXPERIMENT: "CONTAINER CAPTURER"

The first experiment takes place into a rectangular arena with a white surface bounded by a black ribbon. The arena has several containers that can be moving. Each container has a particular colour. The intention of the agent is to remove green containers out of the arena. The background knowledge of the agent is to recognize a container, a green colour and a black ribbon. The actions of the agent are walking inside of the arena, searching for containers, identifying and removing green containers. The motivation of the agent is to continuously remove green containers if it founds different objects. The arena has unpredicted events; and thus: there are some containers moving to different positions on the bounded arena; room light (brightness) may be changed; containers can be horizontal or vertical; and states and objects that are not containers can be used to confuse the agent.

We chose a Lego MindStorm robot (Lego 2002) because it has all sensors and actuators needed to run this experiment. Its body has two traction motors for moving inside the arena, one pressure sensor for sensing containers and two light sensors to detect containers and the black ribbon. Next step, we draw the diagram with actions to be executed by the Lego robot, as seen in Figure 2. The lower layer performs basic tasks, such as backward movement, looking around and forward movement. The upper layer performs laboured decisions about continuously searching for containers, retreating for wrong containers and pushing the specified container out of the arena. Based on Figure 2, the PyramidNet Tool was used to design an ANS project that is shown in Figure 3. ANS project has four learning modules, three sensors, two actuators and respective interconnections.

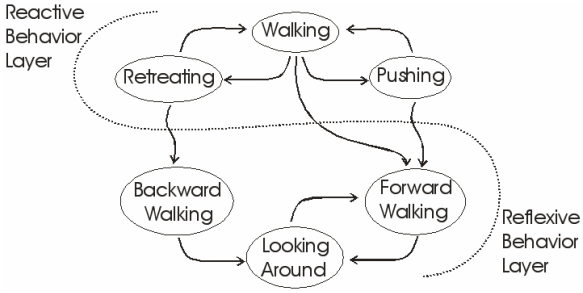

Figure 2: Behaviour Task Plan Diagram

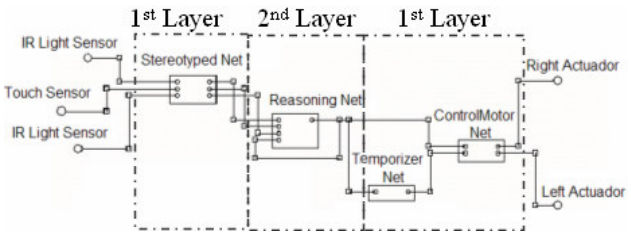

Figure 3: ANS Project

The first learning module, Stereotyped Network, uses FeedFoward with Backpropagation algorithm for reflexive behaviours. It receives signals from sensors and sends data to the second layer. It receives signals from three sensors: two infrared sensors and one touch sensor. The first sensor, infrared, recognizes the container and colour. The second sensor, touch, uses pressure to detect a container and to control the velocity of access. The third sensor, infrared, detects the black ribbon on the floor. The second learning module, Reasoning Net, uses Recurrent networks to make reasoning decisions about detected events in the first layer and transmit to them to the first layer overlapping it. In this case, decisions consist of walking on, retreating on and pushing the green container out of the arena. The third learning module, Temporizer Net, uses Feedfoward ANN to simulate a simple temporizer that triggers turn around movements and back in the first layer when a black ribbon is detected. Finally, the fourth learning module, ControlMotor Net, uses Feedfoward ANN to control the tracking motors for obeying the orders that come from the superior layer. Concluding, a source-code was automatically produced to be applied to the robot. Figure 4 shows the Lego robot selecting a green container and moving it out of the arena.

Figure 4 Lego Arena

This experiment has demonstrated the use of PyramidNet Tool for fast and easy development of applicable ANS. The Lego robot acquired a high degree of reasoning to accomplish goals, test adaptation and ability to deal with unpredictable situations and efficiency in movements.

6. CONCLUSIONS

This paper presented a tool for building Autonomous behaviours in Agents using neural networks as the base to accomplish the reasoning process. The proposed tool – PyramidNet Tool - may be easily used in institutional education because it is focused on common users, without professional knowledge on neural networks and a short agent development time.

The main strengths of PyramidNet Tool are concerned with its graphic design capabilities (including graphical simulation among heterogeneous artificial neural network, sensors and actuators), generation of ready-to-use open-source code and support of several well-known ANN models - Backpropagation for Feedforward networks, Recurrent networks and SOM – Kohonen's maps.

REFERENCES

Fine, T. L. (1999). Feedforward Neural Network Methodology. Calcuta, Springer.

Kohonen, T. (2001). Self-Organizing Maps. Berlin, Springer.

Kolb, B. and I. Q. Whishaw (2005). An Introduction to Brain and Behavior. New York, Worth Publishers Inc.

Lego (2002). Lego MindStorm Hitachi H8: 3804 Robotic Invention System 2.0, http://mindstorms.lego.com/.

Mandic, D., J. Chambers, et al. (2001). Recurrent Neural Networks for Prediction: Learning Algorithms, Architectures and Stability. NY, John Wiley & Sons.

Reinaldo, F. A. F. PyramidNet Tool Project, http://www.inf.ufsc.br/~rei.

Reinaldo, F. A. F. (2003). Projecting a framework and programming a system for development of modular and heterogeneous artificial neural networks. Dept. of Computer Science. Florianópolis, Federal Univ. of Santa Catarina: 86.

Roisenberg, M., J. M. Barreto, et al. (2004). PyramidNet: A Modular and Hierarchical Neural Network Architecture for Behavior Based Robotics. International Symposium on Robotics and Automation - ISRA 2004, Querétaro, México, IEEE.

Wirfs-Brock, R. and R. E. Johson (1990). Surveying current research in object-oriented design. New York, Communications of the ACM.

Wooldridge, M. J. and N. R. Jennings (1994). Agent Theories, Architectures, and Languages: A Survey. Workshop on Agent Theories, Architectures and Languages, 11th European Conference on Artificial Intelligence, Amsterdam, The Netherlands.