INTRODUCTION OF NEURAL NETWORKS TO STUDENTS

Petr Doležel
*** Martin Mariška**

University of Pardubice
Faculty of Electrical Engineering and Informatics
Department of Process Control
Nam. Cs. Legii 565, 532 10 Pardubice, Czech Republic
petr.dolezel@upce.cz

* University of Pardubice
Faculty of Electrical Engineering and Informatics
Department of Process Control
Nam. Cs. Legii 565, 532 10 Pardubice, Czech Republic
mariska.martin@gmail.com

Abstract

The paper deals with the possibility of introducing artificial intelligence and especially artificial neural network methodology to students in an interesting way. To be more specific, the artificial neural network is described through the design of NPC's artificial intelligence in a simple computer game. In the first paragraphs, the methodology of artificial neural networks is described in rather exoteric way through its comparison to biological nervous systems and neural cells. Then, the methodology is used to design a computer game NPC which can observe and learn the behavior of human player. At the end of the paper, the NPC behavior is tested and analyzed, as well as there is mentioned a feedback from students of several education facilities.

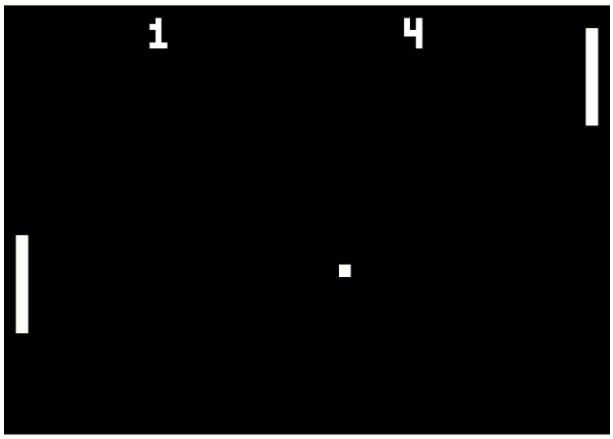
Introduction

Artificial neural networks (ANN) are popular in many branches of science these days and they are even applied to solve various industrial and civil problems. However, education of ANN is limited to universities and, in addition, almost exclusively to optional subjects. The aim of the authors is to hand on information about ANN to a large number of students and, above all, to arouse interest in it. To achieve this objective, a set of popularizing lectures has been drawn up. The extract from these lectures is introduced below.

1 Main Idea

It is not simple to catch today's student's attention for more than five minutes. It is obvious that the lectures should deal with something close to students' interests. Thus, the sphere of computer games is chosen. Almost in every computer game, there is a so-called non-player character (NPC) which should be featured with some level of artificial intelligence. The authors' premise is that it can be remarkably interesting for students to show them the way to do it using ANNs.

Solely for educational purposes, the classical game of Pong is chosen – see Fig. 1. There are two players, each with a paddle and a ball that bounces back and forth between them. Each player tries to position his paddle (which can only move up and down) to bounce the ball back towards the other player. The goal is to make the other player to miss the ball. The aim for students is to design ANN to control the movement of one paddle.



Source: Own

Fig. 1: *Pong game window*

On the other hand, ANNs have been used for NPC control in much more complicated computer games such as Black and White or Colin McRae Rally 2 [1].

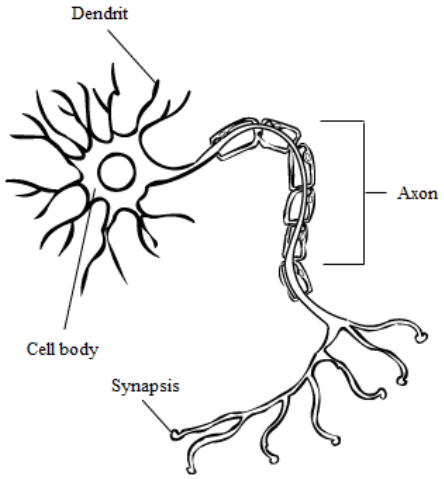
2 The Basics of Feedforward ANN

2.1 Introduction

ANN is really huge methodology nowadays and it is impossible to describe it comprehensively in one paper. In following sections, only basics of ANN are introduced, in correspondence with the amount of information referred to students during popularizing lectures. More detailed description of ANN can be found in [1].

2.2 Analogy to Nervous System

The design of ANN is inspired by analogy with the brain, which is a living proof that fault-tolerant parallel processing is not only possible, but fast and effective, too. Human brain consists of neural cells (neurons). Function of neuron is surprisingly simple in comparison with the whole brain. Each biological neuron consists of a cell body, a collection of dendrites which bring information into the cell and an axon which transmits information out of the cell – see Fig. 2.



Source: Public Domain file from creationwiki.org

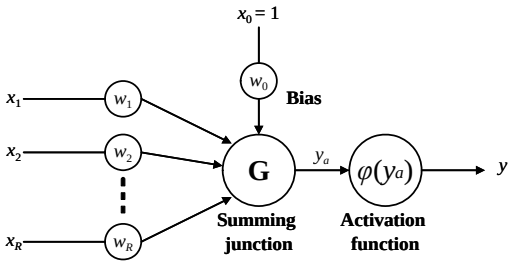
Fig. 2: *Neural Cell*

A neuron produces an output along its axon and the output is a response to collective effect of dendrites incoming to the cell body. The axon from one neuron can then influence the dendrites of other neurons through synapses (some synapses generate positive effect, the other negative one). The processes of learning and remembering are somehow associated to plasticity of the synapses, although this procedure has not been defined in a satisfactory manner yet.

Artificial intelligence (AI) methodology tries to model reasoning processes of human brain. Contrary to some other branches of AI (fuzzy logic, expert systems), ANNs try to model low level functionality of the brain, cell by cell. Thus, the first step is to design an artificial model of the neuron. ANN is then a network of interconnected artificial neurons.

2.3 Model of neuron

In a simplified way, biological neuron is a processor which generates a response to some set of weighted inputs. This process can be approximated by neuron model shown Fig. 3.



Source: Own

Fig. 3: *Artificial neuron*

Three basic elements can be identified here:

- A set of connecting links, each of which is characterized by a weight. Specifically, a signal x_i (which can be either input to the network or output from the previous neuron) connected to a neuron is multiplied by a weight w_i.
- An adder which transforms the set of weighted input signals to one scalar value called activation potential. The operations described in this paper constitute a linear combiner.
- An activation function which processes an activation potential and generates output of the neuron.

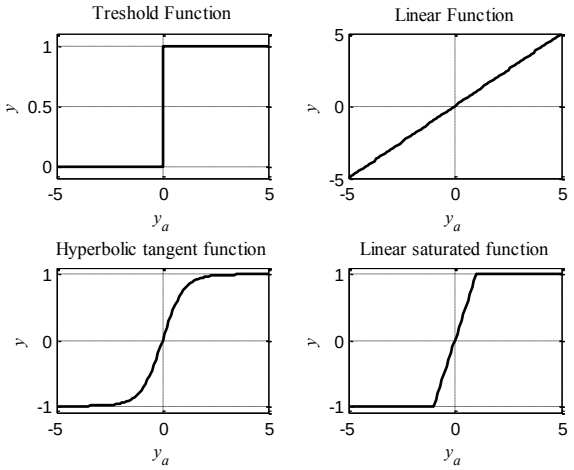
Artificial neuron in Fig. 3 also includes an externally applied bias denoted by $w_0 x_0$. The bias affects the input to the activation function.

Mathematically, the neuron depicted in Fig. 3 can be described by the following pair of equations.

$$y_a = \sum_{i=0}^{R} w_i x_i \tag{1}$$

$$y = \varphi(y_a) \tag{2}$$

Activation function, which is denoted by $\varphi(.)$, can be defined in many ways and its definition crucially affects the response of the whole ANN. In Fig. 4, there are graphs of the most common activations.



Source: Own

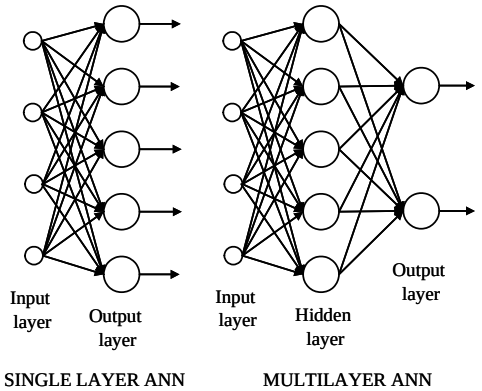
Fig. 4: *Set of activation functions*

The topology of the neuron (number of inputs, definition of adder and activation function) is mostly constant, but the other parameters (especially weights and biases) are tuned through learning.

However, one neuron can solve only simple tasks (most common example is the logic functions approximation). Thus, it is suggested to connect a set of neurons together to build the neural network.

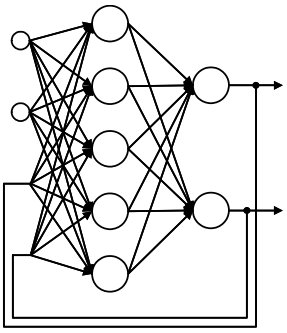
2.4 ANN Topology

ANN consists of one or more neurons connected into one or more layers. Mostly, a layer is compounded of neurons that are not connected to other ones in any manner (there are some exceptions, of course). Examples of ANN possible topologies are shown in Fig. 5 and in Fig. 6 (each node represents one neuron). ANN topology is abbreviated formally by a set of integer numbers – e.g. [2-4-5-1] means ANN with two inputs, four neurons in the first hidden layer, five neurons in the second hidden layer and one neuron in the output layer.



Source: Own

Fig. 5: *Feedforward ANNs*



Source: Own

Fig. 6: *Example of recurrent topology of ANN*

The topology of the ANN depends on a task to be solved. Tasks like function approximation or pattern recognition can be solved using feedforward ANN with one or two hidden layers. On the other hand, recurrent networks are necessary to be used to time series prediction or dynamical processes modeling.

To work properly, weights and biases of each neuron in ANN should be tuned. This process is called learning.

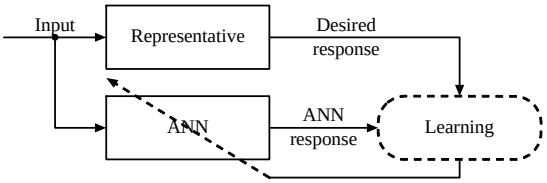
2.5 ANN Learning

The possibility of learning is one of the most attractive features of ANN. Learning means using a set of observations to find optimal (in some sense) values of weights and biases. It is categorized as follows.

- Supervised learning
- Unsupervised learning

For adapting the weights and biases, supervised learning uses a training set of labeled examples, with each example consisting of an input signal and the corresponding response. Unsupervised learning rather implements a task-independent measure of quality of the set of inputs. It is not used as often as supervised learning and it is not discussed here anymore.

To use supervised learning, the first step is to obtain somehow a training set of labeled examples. In other words, a representative to be followed is needed. Then, the process of learning is shown in Fig. 7.



Source: Own

Fig. 7: *Basic diagram of ANN learning*

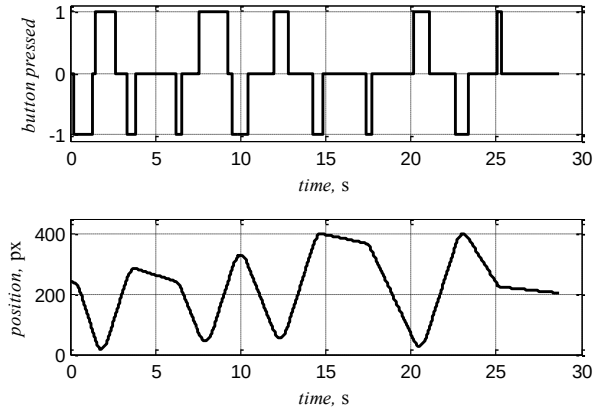
There have been introduced many learning algorithms. Some of them are gradient based (group of backpropagation algorithms [2]), others are not (Levenberg-Marquardt algorithm [3], [4], evolutionary-based algorithms [5]). However, the philosophy is mostly the same. In every iteration of the learning algorithm, the performance of ANN is measured using defined cost function. This information is used then to adapt weights and biases to lower the cost function.

Eventually, frameworks for ANN usage are included in many computing softwares (e.g. Matlab, Statistica) as well as there are several frameworks ready to use in JAVA, C#, C++ etc.

3 Pong Play Using ANN

3.1 Pong Dynamics

The aim of each player playing Pong is not to let the ball leave the game screen. In our particular version each player can press and hold one of the two buttons (one for accelerating upwards and the other for accelerating downwards). The paddle position and speed is affected by the value of acceleration and screen borders. Maximum speed is limited, too. Ball speed is constant. For better illustration, the position course of the paddle is shown in Fig. 8 (y axis of the first graph indicates up-button pressing if the value is 1 and down-button pressing if the value is -1).



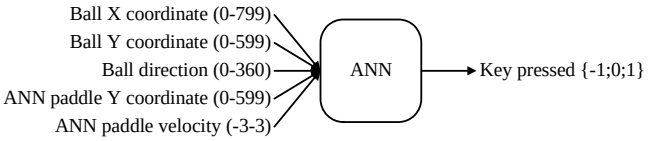
Source: Own

Fig. 8: *Dynamics of the paddle*

3.2 ANN for Paddle Control

To design a proper ANN to paddle control, the first thing to be done is to consider the data which are used by the player to control the paddle successfully.

First experiments are performed with ANN formally shown in Fig. 9.



Source: Own

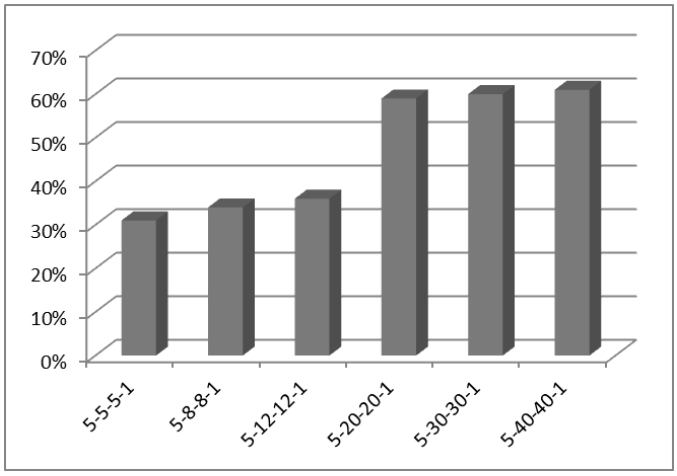
Fig. 9: *ANN for paddle control I*

Training set is simply acquired by a two players´ game. One of the players poses as an ANN teacher while data are collected during every single step of the game (10000 samples are used). Acquired data are standardized and used then for ANN learning using Neural Network Toolbox implemented in Matlab (Levenberg-Marquardt algorithm is used), whereas several topologies are tested. All tested ANNs consist of hyperbolic tangent activation functions in hidden layers and pure linear activation functions in the output layer which is recommended setting [1]. Output of the ANN is rounded to nearest integer from the set { 1; 0; 1} to obtain prospective output.

Testing of the proper behavior is performed in the following way: paddle controlled by ANN is expected to rebound series of balls (100 samples) with random direction (see Fig. 10). Hit ratio is considered then as a performance index.

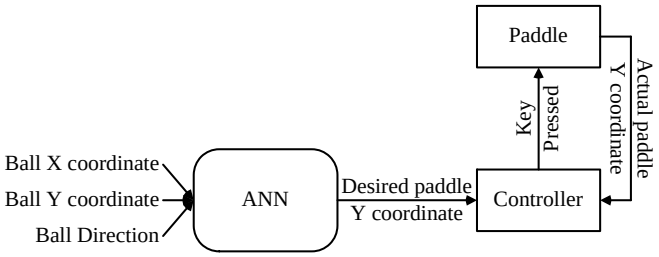
Source: Own
Fig. 10: *ANN testing*

The test results are shown in Fig. 11. It is obvious that the percentage of successful rebounding is too low even for very complex topologies. Thus, another approach should be used.



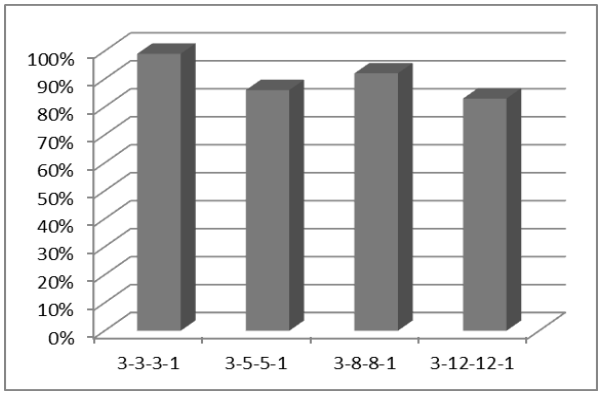
Source: Own
Fig. 11: *Test results I*

After several trials, following composition proved itself as the most suitable – see Fig. 12. This concept probably corresponds more with human player stream of thought – firstly, a player estimates the desired position of the paddle according to the ball dynamics and then he tries to push the paddle into the right position. ANN can also be used as a contoller but in this case discrete version of classical PID controller is used instead because paddle position control is quite a simple linear problem. Learning and testing is performed in the same way as in the previous case and the results are shown in Fig. 13.



Source: Own

Fig. 12: *ANN for paddle control II*



Source: Own

Fig. 13: *Test results II*

4 Popularizing Course Summary

The course is divided into two parts. The lecture introduces the basics of ANN (as it is described in the section 2) and it is followed by the seminar where the students are suggested to design their own ANN for Pong paddle control using specialized software. The summary of the course is resumed in Fig. 14.

4.1 Software for the Seminar

Special software (based on Matlab language) has been prepared for the seminar – see Fig. 15. In one mode, the software can simulate the Pong game for two players while all relevant data are measured. One player is considered here as a supervisor for NPC. In another mode, ANN is designed and trained so that it can approximate its supervisor behavior. In the last mode, NPC's behavior can be tested in the game against human opponent.

Using this software, the students are familiarized with basics of ANNs, they can test how ANN topology affects training results and, last but not least, they enjoy some fun.

The percentage of successful rebounding is almost 100% and the interesting thing is that simple topologies provide higher hit rate. Paddle controlled in this way is a peer opponent to the human player.

Conclusion

The aim of the paper is to suggest the way of introducing ANN to a large number of students in an entertaining way – it means to significantly reduce the theory and to perform some catchy demonstrations. The problem introduced here – Pong paddle control – is decent example for students to start dealing with ANNs. The tasks associated with ANN (data acquisition for training set, topology optimization, …) are well imaginable and the results are well recognizable. The methodology described above has been used with students of two Czech high schools and one university so far and the feedback has been positive.

Literature

[1] HAYKIN, S. *Neural Networks: A Comprehensive Foundation.* New Jersey : Prentice Hall, 1999. 842 s. ISBN 0-13-273350-1.

[2] RUMELHART, D.E.; HINTON, G.E.; WILLIAMS, R.J. Learning representations by back-propagating errors. *Nature*, 1986, No. 323, pp. 533-536. ISSN 0028-0836.

[3] LEVENBERG, K. A method for the solution of certain problems in least squares. *The Quarterly of Applied Mathematics*, 1944, Vol. 2, pp. 164-168. ISSN 0033-569X.

[4] MARQUARDT, D.W. An algorithm for least-squares estimation of nonlinear parameters. *Journal of the Society for Industrial and Applied Mathematics*, 1963, Vol. 11, pp. 431-441. ISSN 0887-459X.

[5] BLANCO, A; DELGADO, M.; PEGALAJAR, M.C. A Real-Coded genetic algorithm for training recurrent neural networks. *Neural Networks*, 2001, Vol. 14, pp. 93-105. ISSN 0893-6080.

PŘEDSTAVENÍ MOŽNOSTÍ NEURONOVÝCH SÍTÍ STUDENTŮM

Článek představuje možnost, jak zajímavou cestou seznámit studenty s problematikou umělé inteligence a především umělých neuronových sítí. Konkrétně, možnosti umělých neuronových sítí jsou uvedeny na příkladu budování umělé inteligence v jednoduché počítačové hře. V první části textu je metodika umělých neuronových sítí obecně popsána a srovnána s biologickými nervovými soustavami a nervovými buňkami. Následně jsou popsané přístupy použity pro návrh inteligentního chování oponenta v počítačové hře tak, aby byl oponent schopen učit se z chování lidského hráče. V závěru je navržená umělá inteligence oponenta testována a vyhodnocena, přičemž jsou také zmíněny ohlasy studentů, kterým byly umělé neuronové sítě tímto způsobem představeny.

VORSTELLUNG DER MÖGLICHKEITEN NEURONALER NETZE FÜR STUDENTEN

Der Artikel stellt eine Möglichkeit vor, wie man auf interessantem Wege die Studenten mit der Problematik der künstlichen Intelligenz, vor allem den künstlichen neuronalen Netzen, bekannt macht. Konkret gezeigt wird die Möglichkeit der künstlichen neuronalen Netze an einem Beispiel, wo man künstliche Intelligenz in einem einfachen PC-Spiel errichtet. Im ersten Teil des Artikels wird die Methodik der künstlichen neuronalen Netze allgemein beschrieben und mit dem biologischen Nervensystem und den Nervenzellen verglichen. Im Folgenden werden Zugriffe beschrieben, die man für einen Entwurf intelligenten Verhaltens des Opponenten im PC-Spiel benutzt hat, und zwar so, dass der Opponent in der Lage ist, aus dem Verhalten des menschlichen Spielers zu lernen. Am Schluss wird der Entwurf der künstlichen Intelligenz des Opponenten getestet und ausgewertet, wobei auch die Reaktionen der Studenten, denen diese künstliche neuronale Netze vorgestellt worden sind, erwähnt werden.

PREZENTACJA MOŻLIWOŚCI SIECI NEURONOWYCH DLA STUDENTÓW

Artykuł przedstawia ciekawy sposób zapoznania studentów z zagadnieniami sztucznej inteligencji, a zwłaszcza sieci neuronowych. Możliwości sztucznych sieci neuronowych są pokazane na przykładzie budowy sztucznej inteligencji w prostej grze komputerowej. W pierwszej części opisano metodologię sztucznych sieci neuronowych, porównując ją z biologicznymi układami nerwowymi i komórkami nerwowymi. Następnie opisano podejście zastosowane do zaprojektowania inteligentnego zachowania przeciwnika w grze komputerowej w taki sposób, aby mógł on się uczyć na podstawie zachowania graczaczłowieka. W zakończeniu zaprojektowana sztuczna inteligencja przeciwnika poddana jest testowaniu i ocenie, przy czym uwzględniono tu również spostrzeżenia studentów, którym w ten sposób zaprezentowano sztuczne sieci neuronowe.