Mathematical Problems of Computer Science 49, 58–65, 2018.

Usage of Neural Networks for ASM Research

Hayk E. Nahapetyan

Institute for Informatics and Automation Problems of NAS RA
e-mail: hayknahapetyan@yahoo.com

Abstract

Purpose of this paper is to describe the possible usage of artificial neural networks for Abelian Sandpile model research. For developing neural networks, Neuroph Studio has been chosen, and Abelian sandpile model has been considered on 2-dimensional grid.

Keywords: Neural Networks, CA, ASM, Neuroph Studio, Limiting shape.

1. Introduction

The structure and function of neural networks (NN) are based on our current understanding of the biological nervous system. NNs are built on a large number of simple and adaptable processing units (PU) which are interconnected in such a way that they can store experiential knowledge through learning from examples and, like biological systems, have the ability to take in hazy information from the outside world and process it without an explicit set of rules. This approach (parallel and distributed) is in contrast to the traditional computing approach which processes information sequentially according to a set of exact rules. Also, their structure and function provide a typical example of the applications of systems perspective concept which puts much emphasis on, in addition to the individual elements and their operations, the relationships among the elements and how they influence each other within the system (Wu, 1992). Perhaps due to some of the difficulties that have been experienced with the traditional expert system applications, and because of the rapid development and introduction of NN system development tools, NNs have created a substantial amount of interest in the manufacturing arena, with systems and techiques being developed for organization, operational, as well as machine-level applications.

The concept of self-organized criticality was first introduced by Bak, Tang and Wiesenfeld in 1987 [1], and gave rise to growing interest in the study of self-organizing systems. Bak et al. argued that in many natural phenomena, the dissipative dynamics of the system is such that it drives the system to a critical state, thereby leading to ubiquitous power law behaviors. This mechanism has been invoked to understand the power law distributions observed in turbulent fluids, earthquakes, distribution of visible matter in the universe, solar flares and surface roughening of growing interfaces. The Sandpile models, being a class of cellular automata, are among the simplest theoretical models, which exhibit self-organized criticality. A special subclass of interest consists of so called Abelian sandpile models (ASM). The Abelian property means that the final stable state of the CA is independent of the order

in which the updates of cells are carried out. This property plays a key role during the numerical, as well as analytical studies of the ASM [2]–[5]. In this paper we describe the usage of neural networks in the research of Abelian Sandpile model on 2-dimensional grid. The problem statement is considered in the "Problem of interest" section.

2. Basic Structure of NNs and Neuroph Studio

As mentioned above, artificial neural networks consist of processing units (PU), which are the building bricks of NNs. PUs usually take the form shown in Fig.1, and emulate (assumingly) the function of a neuron in the brain. Basically, PUs are logic processing devices endowed with a fundamental function over the sum of their weighted inputs and a certain threshold value. Mathematically, this is expressed as:

$$y_i = f_i\left(\sum_{j=1}^{n} w_{ij}x_j - s_i\right) = f_i(a_i), \tag{1}$$

where y_i, w_{ij}, x_j and s_i stand for the PU output signal, the weight of the j to i interconnection, input value from PU_j and the threshold value of PU_i, respectively. The input to PU_i can be either the output from other PU_s or directly from outside the NN, i.e., input to the NN. The output from PU_i can be used either as an input to the subsequent PU_s or as an output from the NN. The value of w_{ij} determines how strongly the output of PU_j influences the activity of PU_i. The magnitude of a weight can be changed over time. During a training operation, it is mainly through this mechanism that the PU is made adaptive to new information put to it, and the learning process is accomplished. As will become clear later, the total weight matrix W of an NN encompasses and reflects the $NN's$ knowledge and skills that it has learnt through previous training, and is therefore referred to as its long-term memory.

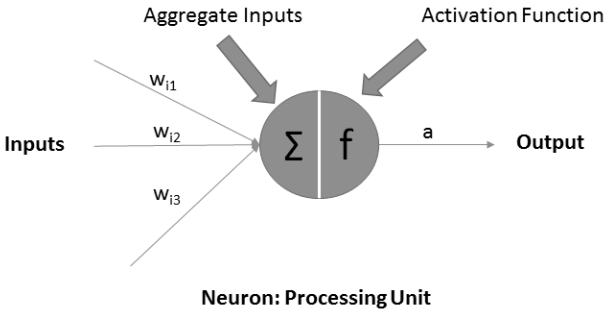

Fig 1. The processing unit.

The threshold s_i acts as a filter for incoming signals. The term inside the brackets in Equation 1, a_i, is known as the activation of PU_i, which provides temporary and local information around it. This is therefore referred to as the $PU's$ short-term memory.

The value of a_i is transformed by the $PU's$ output function, f_i, to determine the magnitude of its current output signal. A number of activation functions have been used to

construct NNs, of which the step function is the simplest and the most straightforward (Fig. 2a). With the step activation function, a PU produces an output signal of either $'1'$ or $'0'$ depending on whether or not the level of its activation is above a certain threshold value. That is:

$$y_i = \begin{cases} 1 & \text{if } a_i > 0 \\ 0 & \text{otherwise} \end{cases}$$

where

$$a_i = \sum_{j=1}^{n} w_{ij} x_j - s_i$$

However, in order to filter out the noise and hence enhance the ability of achieving a true steady state of operation, for some NNs a sigmoid activation is usually used in practice, expressed in the form of:

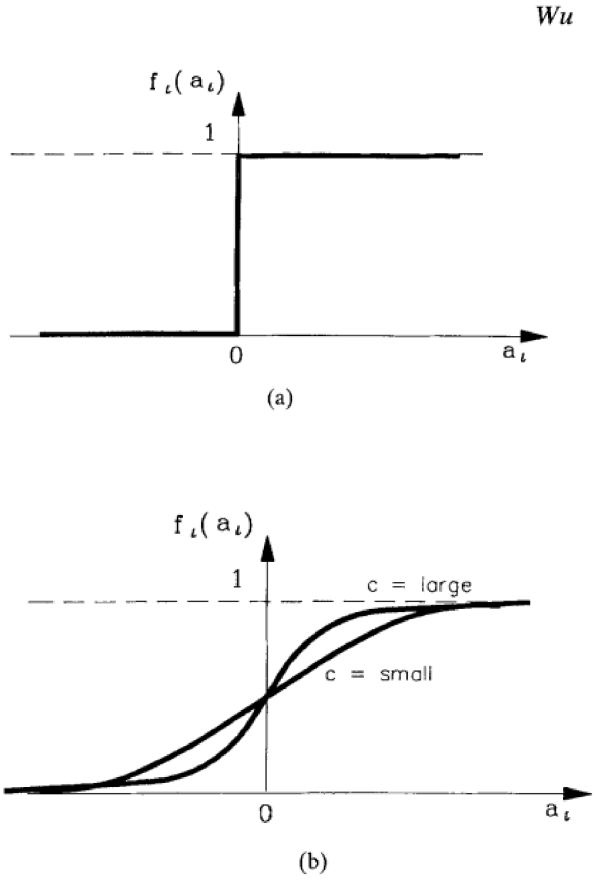

Fig. 2. Activation functions: (a) step activation function; (b) sigmoid activation function.

$$y_i = f_i(a_i) = \frac{1}{1 + e^{-ca_i}},$$

where,

$$a_i = \sum_{j=1}^{n} w_{ij} x_j - s_i.$$

Here c is a constant which determines the degree of 'uncertainty' introduced into PU_i activity.
The general shape of this function is as shown in Fig. 2b. Some of the advantages offered by this type of function will become clear later in the text ($1/c$ is also known as the 'temperature'). In some cases its value can be set at an artificially high level initially to 'shake' the NN so that it has a better chance of achieving its true stable state. This is then gradually reduced to allow it to cool down to the ideal state, i.e. step function with zero degree of temperature. This is known as simulated annealing).

Neuroph Studio is lightweight Java neural network framework to develop common neural network architectures. It contains a well-designed, open-source Java library with a small number of basic classes, which correspond to basic NN concepts. It also has a nice GUI as neural network editor to quickly create Java neural network components. For creating and testing neural networks over cellular automata, Neuroph Studio has been chosen.

3. Sandpile Model

Consider an undirected graph $G = (V, E)$ described with the set of vertices $V = \{v_1, v_2, \ldots, v_N\}$ and the set of edges E. Each vertex $v_i \in V$ is assigned a variable h_i, which takes integer values and represents the height of the sand at that vertex. h_i^{max} denotes the maximal allowed height for the vertex v_i in the graph G. For a d-dimensional lattice, we take $h_i^{max} = 2d + 1$. C_T denotes the set of heights h_i, which determines the configuration of the system at a given discrete time T. A configuration is called stable, if all heights satisfy $h_i < h_i^{max}$. The vertex v_i is called closed, if $h_i^{max} = deg(v_i)$, where $deg(v_i)$ indicates the degree of v_i. The dynamics of the system is defined by the following rules. Consider a stable configuration C_T at a given time T. We add a grain of sand to a random vertex $v_i \in V$ by setting h_i to $h_i + 1$ (we assume that the vertex is chosen randomly with a uniform distribution on the set V). This new configuration, if stable, defines C_{T+1}. If $h_i \geq h_i^{max}$, then the v_i becomes unstable and topples losing h_i^{max} grains of sand, while all neighbors of v_i receive one grain. Note that if the vertex is open, then the system loses grains. During the toppling of the closed vertices, the number of grains is conserved. Note also that toppling of a vertex may cause some of its neighboring vertices to become unstable. In this case, those vertices also topple according to the same toppling rule. Once all unstable vertices are toppled, a new stable configuration C_{T+1} is obtained. If the finite connected graph G has at least one open vertex, then all vertices become stable after a finite number of topplings. Moreover, the new stable configuration is independent of the toppling order. Let $\hat{a}_i$ be an operator, which acts on sandpile configurations and adds a grain to vertex i. It can be easily shown that $\hat{a}_i \hat{a}_j = \hat{a}_j \hat{a}_i$. This is the reason why the sandpile model is called Abelian.

4. Problem of Interest

The research problem concerns the Abelian Sandpile model over a 2 dimensional square lattice of size $(n * n)$, where n stands for the number of nodes on each lattice line. Consider $n- > \infty$. $r_{i,j}$ will be called the distance between V_i and V_j, or in other words, the minimum count of edges, which is needed to pass between V_i and V_j. $C_{i,j}$ will be the min count of grains, that by toppling that much grains on V_i, and letting the model to become stable, we can make sure that at least one grain has reached the node V_j, in oder words $h_j \geq 1$. Let's consider a 2-dimensional lattice, where $h_i = 0, \forall i, 0 \leq i \leq n^2$. Let's choose any V_i node on the lattice. The problem is to find a formula describing connection between $r_{i,j}$ and $C_{i,j}$. There are articles [6], [7] regarding this problem. It should be noted that an exact formula describing $C_{0,J}$ dependency on j does not exist, also all the results obtained up to date are interpreted via approximation formulas only.

In order to obtain correct results for $C_{0,j}\,, where j > 0$, a software program has been created which simulates the sandpile model and produces the data for neural network learning.

4.1 Results

In this subsection we give a comparative analysis of the results obtained by the applications of newly created software package and the ones that gave neural networks. As an example of a neural network, a so-called "multilayer perceptron" with one input neuron and one output neuron, has been chosen. Bias neurons have been used in NN structures, where the sigmoid type has been chosen for transfer function, meanwhile, the learning rule is developed based on back-propagation methodology. In listed examples, the difference between neural networks structure are hidden neurons count only. There are 3 cases regarding the neural networks architecture depending on neurons' count. The first case concerns the presence of a lot of neurons, when the NN will memorize all input values while training and will produce the results of tests without thinking. In the second case, a very little number of neurons are under consideration, and the NN will produce rather wrong results and, therefore, will be not that smart. The third case, named "The Golden Mean", has a big range regarding the neurons' count, and it is not that hard to find out the structure corresponding to this case. In this example, the results of tests, given below in Figures 4 and 5, illustrate that the total mean square error is less for NN with 10 neurons comparing with the test results with 14 ones. The bigger the training set is, the more neurons are needed.

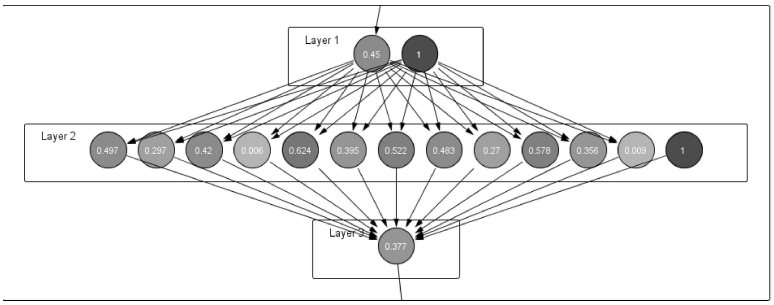

Fig. 3. Neural Network state after training

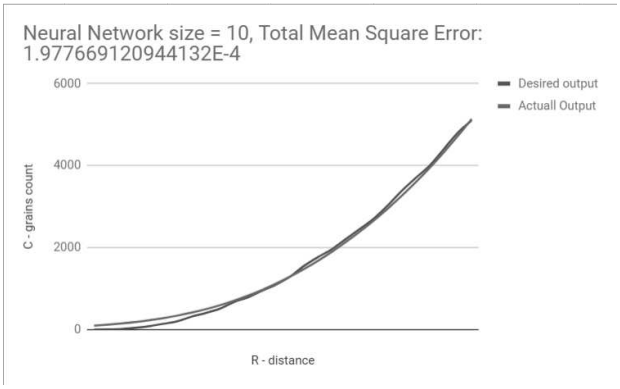

Fig. 4. Test results with 10 neurons

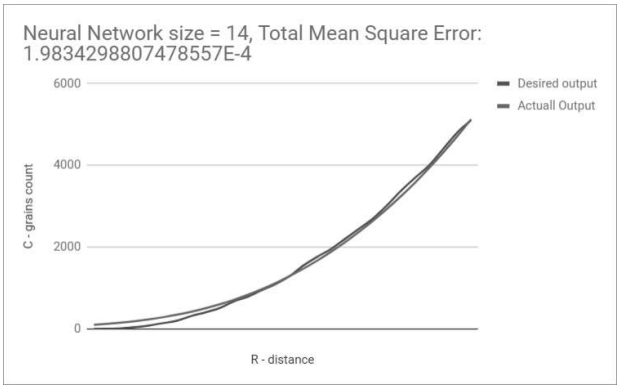

Fig. 5. Test results with 14 neurons

Avarage Solidity: 1
Layer Solidity:
Critical Solidity:
Is Recurrent: False
Is Stable:True
Nodes with height 1: 0
Nodes with height 2: 12
Nodes with height 3: 24
Nodes with height 4: 13
Nodes with height 5: 0
Nodes with height 6: 8
Nodes with height 7: 0

Fig. 6. Attributes

5. Conclusion

In this paper, Neural Networks' usage for solving ASM problems has been discussed. The goal of this work was to find out neural network structure corresponding to the problem such as the one described in "Problem of Interest" section. Comparative analyzes between actual results and the ones that gave NN has been discussed, and in case of acceptable oversight ASM simulation could be changed via neural networks described in this research in order of minimizing time consumptions. Also the comparative analysis of different neural networks' architectures has been conducted. Perspectives on work are to use cluster systems for getting results of $C_{0,j}$ for bigger j and compare with results of already known solutions/formulas and with the ones from ASM simulation.

6. Acknowledgement

The author is grateful to Dr. S. Poghosyan and Dr. Y. Alaverdyan for important discussions and critical remarks at all stages of the work. This work was supported by the State Committee of Science MES RA, in the frames of the research project No. 16YR-1B008.

References

[1] P. Bak, C. Tang and K. Wiesenfeld,"Self-organized criticality: An explanation of the 1/f noise",*Phys. Rev. Lett.,* vol.59, no. 4, pp. 3811–7384, 1987.

[2] V. S. Poghosyan, S. Y. Grigorev, V. B. Priezzhev and P. Ruelle, "Pair correlations in the sandpile model: A check of logarithmic conformal field theory", *Phys. Lett. B,* vol. 659, pp. 76817772, 2008.

[3] Su. S. Poghosyan, V. S. Poghosyan, V. B. Priezzhev and P. Ruelle, "Numerical study of correspondence between the dissipative and fixed-energy Abelian sandpile models", *Phys.Rev. E, 84, 066119*, 2011.

[4] V. S. Poghosyan, S. S. Poghosyan and H. E. Nahapetyan, "The Investigation of Models of Self-Organized Systems by Parallel Programming Methods Based on the Example of an Abelian Sandpile Model", Proc. CSIT Conference 2013, Yerevan Armenia, Sept. 23-27, pp. 260-262, 2013.

[5] H. Nahapetyan, J.-Pierre Jessel, S.Poghosyan and Y. Shoukourian,"A Multi User and Multi Purpose CA Simulator17",*Phys. Rev. Lett.,* vol.59, no. 4, pp. 38117384, 1987. Proc. CSIT Conference 2017, Yerevan Armenia, Sept. 23-27, pp. 260-262.

[6] L. Levine and Y. Peres, "Asymptotics for Rotor-Router Aggregation and the Divisible Sandpile", Potential Anal (2009) 30: 1. https://doi.org/10.1007/s11118-008-9104-6

[7] A. Fey and F. Redig, "Limiting shapes for deterministic centrally seeded growth models", *J. Stat. Phys.*, vol. 130, no. 3, 57917597, 2008.

Submitted 06.10.2017, accepted 22.12.2017.

Նեյրոնային ցանցերի օգտագործումը ավազակույտի աբելյան մոդելի հետազոտության համար

Հ. Նահապետյան

Ամփոփում

Այս հոդվածի նպատակն է նկարագրել արհեստական նեյրոնային ցանցերի հնարավոր օգտագործումը ավազակույտի աբելյան մոդելի հետազոտության համար: Նեյրոնային ցանցերի մշակման համար ընտրվել է Neuroph Studio-ն, իսկ ավազակույտի աբելյան մոդելը դիտարկվել է երկչափ քառակուսային ցանցի վրա:

Использование нейронных сетей для исследования модельи песчаной кучи

Г. Нагапетян

Аннотация

Цель этой статьи - описать возможное использование искусственных нейронных сетей для исследований модельи песчаной кучи. Для разработки нейронных сетей была выбрана Neuroph Studio, а модель песчаной кучи была рассмотрена на двумерной сетке.