International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 05 Issue: 01 | Jan-2018 www.irjet.net p-ISSN: 2395-0072

AI Neural Network Disaster Recovery Cloud Operations systems

Dr Lenny Michael Bonnes

Colorado Technical University, Colorado Springs. Colorado

Abstract – AI *Neural networks can meet the needs of Disaster recovery and Systems and security in cloud production systems. Through constant monitoring of the network applied as a repair and redeployment model. In fact, if a system fails to operate the supporting neuron nodes can rebuild the system while establishing new connections. Neural networks are an effective method for forming control policies in difficult decision problems. Building a neural network with a layered approach to an organizations cloud production network allows the use of cooperative coevolutionary algorithms and canonical evolutionary algorithms to support and automate systems, as a self-remediation tool.*

Key Words: **Neural Networks, Cloud, Disaster recovery, Cooperative coevolutionary algorithms, canonical evolutionary algorithms, Distributed problem solving,**

1.INTRODUCTION

Neural computing originates from biological processes, it will be useful to review the properties of a biological neural network at a summary level, before introducing Artificial intelligence remediating disaster recovery neural network system. Our human brain deals with pattern matching and pattern handling with ease and efficiency that no computer system can match it performs these tasks not in a consecutive fashion but in a parallel fashion using a large network of neurons (or nerve cells). The brain is organized in a hierarchy with successively higher layers performing more complex and abstract operations on the input data. Each layer performs its processing of the input data before passing the result up to the next layer. The neurons for the "processing elements" of the brain. The human neocortex contains over 10 billion neurons with each neuron connected to thousands of other neurons.

The brain contains a vast number of different neurons each with a marginally differing structure and properties. Much like the designs of neurons in the AI neural network DRCO systems structures and properties vary.

In the current Artificial Intelligence (AI) neural models they remain simplistic when compared to the brain, and use a simple summation and threshold device as the basic processing element in a layered network. Artificial neurons are the fundamental building blocks of neural networks. A neuron takes a set of inputs X; which are equivalent to the excitation or the inhibition signal levels of a neuron.

A typical artificial neuron has a single output and several inputs, usually one from each neuron in the preceding layer. These inputs are then acted upon by a set of associated weights W; which correspond to the synaptic strengths of a neuron. These weighted inputs are then compared with a threshold value and the output is delivered depending on the result of thresholding. The weighting factors are analogous to the synaptic strengths within a brain. The artificial neurons are usually connected in a simple layered structure. Most neural network models consist of either two or three layers of neurons since it has been shown that any continuous mappings can be achieved by a three-layered system a multilayer network where each input X; to a neuron Nj, in a layer has an associated weight W; j. The outputs from each layer are then broadcasted as inputs.

1.1 AI Self –Disaster recovery monitoring system for cloud operations.

This AI self-remediating disaster recovery monitoring system for cloud operations (AI DRCO systems) is structured holistically much like the human brain. The core response server is built to manage the activities of the network neuron collective. And supporting servers. Forward security attack pattern recognition severs, (FSAPR) the FSAPR supplies the pattern identification to the core responsive server. Parietal network logic end server (PNLE)network design changes and recognition and FSAPR stimulation. Temporal log pattern preceptor (TLPP)server supporting FSAPR server and parietal network logic server. The brain neocortex equivalent is the network neuron collective.

1.2 Perimeter Network distributed problem solving (DPS), agents

Guarding the perimeter of a network using a distributed problem solving (DPS) agents; DPS is the cooperative solution of a problem by a grouping of loosely coupled agents resulting in a decentralized computational model. There are no centralized data stores, and no agent has enough information to make a complete decision; an agent requires assistance from other agents in the decision-making process. DPS agents are distributed over the entire cloud computing environment. fundamental areas of interests of the DPS revolves around the distribution and coordination of computation among its society of agents so that structural demands of the task are matched.

The DPS node resolves to a manager node which announces a task for which multiple eligible perimeter nodes respond with an offer. An offer/task is presented by a level of need and correction to the network nodes view/state. Each perimeter node will receive instructions after a negotiation process, and in case a perimeter node requires assistance with its current problem that to solve as part of the original request.

In this quest for answers, the perimeter node will assume the role of manager node the node and send a sub-request the part of the problem it needs help to solve. The methodology for resolution for a task is specified in the programming of the perimeter node.

2. Distributed Cloud Multinet Networks

How does this neural network model effect a distributed cloud multinet network? AI neural network DRCO systems leverage a combination of several networks within a single network modular by design that cooperates in solving a given offer/ task:

- multinet networks can perform more complex tasks than anyone subcomponent or system.
- Multinet networks are robust by nature of design than a single network.

Deploying a neural network on a multinet network can build a neuron collective.

The network neuron collective is designed to independently and sequentially calculate the needs of the network by node interactions within Cloud multinet Networks. The neural network DRCO systems will live within the foundational architecture of a production network. This production multi-network framework presented has been designed for an evolving neural network. the support and uses benefits of a neural network DRCO system is built on two different paradigms: Cooperative coevolution and multi-objective optimization. Design of a neural network collective is as such that numerous decisions are calculated on the network nodes that have a significant impact on the performances of the offer or task. Decisions that must be faced when designing a AI DRCO System neural network:

- Design a model of creating and training the individual network neuron nodes.
- Combining the network neuron nodes, and the mechanism for gathering the different weights for each network neuron node and subsystems and subnets.
- Use of multi-objective optimization algorithm designed for performance measurements of each network neuron node.
- The inclusion of a standardized build generating technique using multiple independent phases: Model generation and design combination. Interactions among the trained networks are exploited at the combinatory phase.

However, cooperative coevolution algorithm allows us to obtain additional sub-networks without introducing several relationships that can bias the learning process and bring about the improvement of the collaborative features of the AI neural network DRCO systems as a collective model. While this Neuron Network Collective model presents a multi-objective optimization model the first objective is the evaluation of the fitness of the neural network DRCO systems, and the performance of the Neuron Network Collective is an important aspect of an integration process within the AI neural network DRCO production networks. Fitness evaluations require an evaluation of different objectives for each network neuron collective. This process repeats in each network and sub-network, each network has its own neuron collective.

While the neuron network collective provides a multi-objective evaluation of a network, this evaluation process improves the following features in the design of the network neuron collective.

- Provisioning of performance measurements of the individual networks. Which can include the evaluation of the performance in the sub-networks from different points of observation.
- The multi-objective evaluation methods for encouraging diversity among the members of network neuron nodes. All the while measuring the diversity of the node network.
- Provide standardized neuron node and calculate an objective evaluation to reduce the complexity to the network neuron nodes without biasing the evolutionary process.

2.1 Network Neuron Collective

The design of neural networks over the last few years has used a standard evolutionary computation model; this is a set of global optimization techniques like swarm theory. Past research has found evolutionary computation is ideal for training and developing neural networks. Cooperative coevolution is ideally based on the evolution of coadapted subcomponents without external stimulation. In cooperative coevolution, the number of nodes evolves at the same time. In AI DRCO system design, cooperative coevolution approaches its design in a modular system. The cooperative coevolutionary model will offer a natural way for the evolution of the cooperative parts. It is with this model that the case of cooperative Neuron Network Collective, will exist within AI DRCO systems where the accuracy of the individual networks is not enough to assure a good selection. There is cooperation among different networks that improve the performance selection significantly. The network neuron

collective model is based on two separate populations that evolve cooperatively. These two populations are the following.

The population of neuron Nodes: This population consists of several independent subpopulations within a production network. The evolution of subpopulations of networks is an effective way of keeping the networks of different populations diverse. The absence of inherited material exchange among subpopulations produces a more diverse network this combination is more effective every subpopulation is evolved using evolutionary programming. Network neuron collective: each node of the population of the collective is a collectively formed by a network from every network subpopulation. Each network has an associated weight. The population of collectives keeps track of the best combinations of networks, by selecting the subsets of networks that are promising for the final collection.

The two populations, the network neuron collective and the population of neuron nodes, evolve cooperatively. Each generation of the whole system consists of a generation of the network population followed by the generation of the collective population. The learning process of the networks must consider the cooperation among the networks for obtaining better collectives. Cooperation among the subpopulations in a cooperative coevolution helps develop the learning process, it is necessary to force cooperation to assure good results. This forced cooperation is done by giving the neuron nodes several objectives for each network the objectives will assist to encourage cooperation. Additionally, every network is subject to backpropagation training [3] through the evolution with a certain probability. This allows the network to learn from the training set. The back-propagation algorithm is implemented as a mutation operator. The decision process must be made to design a collective of neural networks.

2.2 Network Neuron Collective

Using a generalized multilayer perceptron, this GMLP consists of an input layer, an output layer, and several hidden neuron nodes interconnected among them.

Figure -1: Generalized Multilayer Perceptron

- $x_i = X_i, h_{i=\sum_{j=1}^{i-1} w_{ij} x_j}$
- $x_{j=f(h_j)}\ Y_{i=x_{i+m+N}}$

Where $(w_{i\ j})$ is the weight of the connection from node j to node i. The representation of a GMLP is seen in figure 1 as you see the ith node, is not an input node, it does have connections from every jth node j< i. The main reason to using GMLP is the evolution of networks. The structure allows the definition of complex surfaces with fewer nodes than a standard multilayer perceptron with one or two hidden layers. Forming the network population by using N_p subpopulations. While using subpopulations with a fixed number of networks it is necessary to codify the subpopulations. Such as $x_i = X_i$, $1 \leq i \leq m$. The subpopulations are not fully connected. When a network is initialized, each connection is created with a given probability. The population is subject to operations of replication and mutation. The algorithm for the evolution of the subpopulations of networks is like other evolutionary algorithms such as GNARL [4] or EPN [5] The steps for generating the new subpopulations:

Networks of the initial subpopulation are created randomly. The number of nodes of the network h is obtained from a uniform distribution $0 \leq h \leq h_{max}$ Each node is created with several connections c taken from a uniform distribution $0 \leq c \leq c_{max}$.

The new subpopulation is generated replicating the best P% of the previous subpopulation. The remaining (1- P) % is removed and replaced by mutated copies of networks selected by randomized selection from the best P% individuals.

There are two types of mutation within the neural network design: parametric and structural. The severity of structural mutation is determined by the relative fitness, F_r $(V) = e^{-\alpha F(v)}$ Where F (V) is the fitness value of network v, and $\propto$ is a parameter that must be chosen. Parametric mutation consists of the modification of the weights of the network without changing its topology. Many parametric mutation operators have been suggested in differing neural network literature: such as random modification of the weights, simulated annealing, and backpropagation, among others.

Parametric mutation comprises the modification of the weights of the network without modifying its topology. Many parametric mutation operators have been suggested in varying literature: random modification of the weights, simulated annealing, and backpropagation.

In this research, the mutation operator uses a back-propagation algorithm and is performed for a few iterations with a low value of the learning coefficient η. Parametric mutation is carried out after the structural mutation. Structural mutation displays complexity in the modification of the structure of the network. While retaining the behavioral link between parents and their offspring this link avoids generational gaps that can produce inconsistency in the evolution algorithm. There are four different structural mutations that will be applied to the network neuron collective

- Addition of a node: The node is added with no connections to enforce the behavioral link with its parent.
- Deletion of a node: A node is randomly selected and deleted together with its connections.

- Addition of a connection: a connection is added, with weight 0 to a randomly selected node. There are three types of connections: from an input node, from another hidden node to the output node. Each connection of each type may end up biased.
- Deletion of a connection: a connection is selected, following the same criterion of the addition of connections, and removed.

Table -1: Parameters of network structural mutations

Mutation	Δ_m	Δ_M
Add node	0	1
Delete node	0	2
Add connection	1	4
Delete connection	0	3

All the above mutations can be made in a single mutation operation over the network. For each mutation, there is a minimum value Δ_m and a maximum value Δ_M. The number of elements (nodes or connections involved in the mutation is calculated $\Delta = \Delta_m + F_r(\mathrm{v})(\Delta_M - \Delta_m)$ prior to creating a mutation, the number of elements Δ is calculated If $\Delta = 0$, The mutation is not executed. The values of network mutation parameters used *table 1* for the initialization of the weights of the networks, weights should be chosen randomly but in such a way that the sigmoid is primarily activated in its linear region. If weights are all very large, then the sigmoid will saturate resulting in small gradients that make learning slow. If the weights are very small, then gradients will also be very small. Intermediate weights that range over the sigmoid linear region have the advantage that the gradients are large enough and learning can proceed, and the network will learn the linear part of the mapping. Achieving a balanced gradient range requires coordination between the training set normalization, the choice of sigmoid, and the choice of weight initialization. There is a requirement that the distribution of the outputs of each node have a standard deviation (σ) of 1. This a normalized training set designed to achieve the standard deviation close to 1 at the output of the first hidden layer the use of this sigmoid $f(x) = 1.7159$ tanh $(\frac{2}{3}x)$ with the use of this sigmoid the properties $f(\pm 1) = \pm 1$ the second derivative is maximum at $x = 1$, "the gain is close to 1". This symmetric sigmoid requires the avoidance of initializing with very small weights. Adding a small linear term to this sigmoid may help avoid the flat regions. Such as I.e., $f(x) = tanh(x) + ax$, using a uniform distribution with the interval $-3/\sqrt{m}$ and $3/\sqrt{m}$ where m is the number of inputs to the network. A single generation of the evolutionary process for the network h is illustrated this illustration shows the possible evolution of a network neuron collective within a single generation.

2.3 Network Neuron Collective Search

Stochastic evaluative search is a scheme which can be written compactly as $\emptyset^{i+1} = S.\emptyset^i$, where $\emptyset^i$ is a solution $\emptyset^{i+1}$ is a successive solution and S is a transition function that utilizes an evaluation $\psi(\phi^i)$ of ϕ^1. This algorithm is a sample of an approximate solution. Much like the simulated annealing. Simulated annealing works with a single solution while evolutionary algorithms work with a population of solutions, on the opposite side of the search space is evolutionary algorithms that tend to be more resistant to being trapped in local minima. Since this project will use an evolutionary optimization and evolutionary optimization is population-based there is a higher chance that multiple promising subregions of the space X will be simultaneously considered during the search. As this evolutionary optimization, will be applied to neuron network collectives the search must include subnets and neuron nodes within that subnet.

2.3 Configuration of the network neuron collective environment

The configuration of the network neuron collective consists of several logical modular clusters of network nodes and a behavior neuron recording node. Nodes in a logical cluster correspond to a class of functionally equivalent entities and are distributed over a network. Nodes in a logical cluster correspond to a class of functionally equivalent entities and in general, are distributed over a network. In this environment, three logical clusters of the network are developed.

- DPS nodes: Each node sits on the perimeter of the networks; each neuron node corresponds to firewalls and load balancers and endpoints.
- System update neuron nodes: Each system node sits on the perimeter of the networks; each neuron node corresponds to the perimeter distribution neuron, providing update searches.
- Behavior neuron nodes: behavior neuron node sits on the perimeter of the networks; each neuron node corresponds to the perimeter distribution neuron; the behavior neuron node is part of the subpopulation.

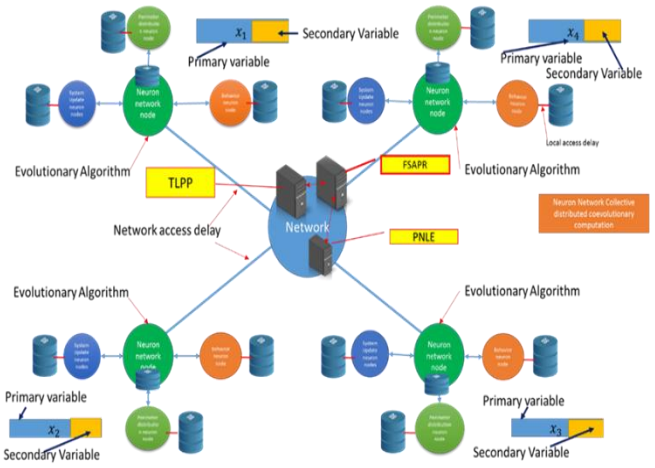

Fig -1: Name of the figure

2.3 AI DPS system node design

Applicability of alternative optimization algorithm. This section presents an evaluation of the applicability of alternative optimization algorithms to the autonomous security network agent. Given that the problem class has multiple variables, constraints, and nonlinear objectives, the goal is to select an optimization algorithm that can tackle this broad class of integrated planning decision problems that may be combinatorial in nature.

Rationale

Combinatorial problems incur a heavy penalty due to dimensionality since the number of options grows exponentially with the size of the inputs. Most combinatorial optimization problems are NP-Hard constraints

Constraints

The problem model represents a set of linear equality constraints restricting assignment of the decision variable Ø_ijk over the index space (I = 1,2,3…, p), (j = 1,2,3…, s), (k = 1,2, 3…, m). A linear equality constraint set written AØ=b, where Ø is a vectorization of the decision- variable vector, and has a length l_c = (p*s*m), A is a l_r ⟦xl⟧ _c matrix comprising entries from the set {-1,0,1} and b is an l_r – dimensional column vector. Including constraint (3.7) which defines a workable space given by {x = {Ø: AØ=b, Ø∈ Z^ (l_c) +} which convex.

In this model of computation, the evolutionary algorithm at any node i performs an evolutionary search based on its primary variable block ⟦ x ⟧ _i using local and rapidly accessible information, and it is assumed that accessing the interconnection network for purposes of communication between the nodes is delay prone, and so each node must perform many local computations. Between cycles in the interest of efficiency. Within the network, there can be relative computational delays of the centralized and distributed algorithms when they are implemented in a network environment.

3. Agents design neural networks

The DPS agents start the evolutionary search by searching each of the perimeter nodes for available decision resources and encoding references to the discrete choices in the appropriate one or two or more alternative forms of mutations or changes found at the same place on the nodes. sets of its representation data structure. A certain mutation of the network will set corresponding to a certain part node type stores an encoded list of equivalent parts available for spin up. All perimeter nodes act in a similar fashion to mutations in the networks x_iIn this model of computation, the evolutionary algorithm at any node i performs an evolutionary search based on its primary variable block x_i using locally accessible information, it is assumed that accessing the interconnection network for purposes of communication between the nodes is delay prone, and so each node performs large number of local computations between communication cycles. Given space x and a variable distribution, each node i performs a local evolutionary search in its primary subspace x_i while the variables corresponding to the secondary subspaces at a node are secured. An intercommunication operation updates the respective secondary variables at all nodes. Following this, the local search proceeds using updated information, and the local and global operations of the distributed search alternate, resulting in a cooperative search. The search space x in this computation is the product space of the p subspaces and is given by $x = \prod_i^p = 1$

4. CONCLUSIONS

The applicability of AI Self–Remediating Disaster recovery monitoring system for cloud operations. Can improve the disaster recovery of many businesses that rely on cloud computing. Many times, production and security lack the personnel to patch update and recover systems due to the lack of knowledgeable employees. This AI Self – Remediating Disaster recovery monitoring system for cloud operations. Fills the gap between HR resources and system resources.

REFERENCES

[1] C. H. M. D. O. B. Nicolas Garcia Pedrajas, "Cooperative Coevolution of Artificial Neural Network Ensembles for Pattern Classification.," IEEE transactions on evolutionary computation, vol. 9, no. 3, p. 32, June 2005.

[2] J. W. Z.H. Zhou, "Ensembling Neural networks: Many could be better than all," Artificial Intelligence, vol. 137, no. 1-2, pp. 239-253, May 2002.

[3] P. J. Werbos, The roots of Backpropagation: from ordered derivatives to Neural Networks and Political Forecasting, New York: Wiley, 1994.

[4] X. a. Y.Liu, "A new evolutionary system for evolving artificial neural networks," IEEE Trans. Neural Networks, vol. 8, pp. 694-713, 1997.

BIOGRAPHY:

Dr. Lenny Mike Bonnes is a computer scientist and an expert in Data Security currently a Chief Information Security officer