RC Research Fellowship Proposal: OpenBrain

William Guss
26793499
wguss@berkeley.edu

January 3, 2016

1 Background

Machine learning research in the past decade has had a large focus on variations of the artificial neural network (ANN), a biologically inspired algorithm originating from the mathematical neuron model proposed by McCulloch and Pitts[2], the perceptron neuron proposed by Rosenblatt[3], and made practical by the back propogation algorithm created by Werbos [4]. Although derivative of biological neural networks, the ANN and its variations stray away from many features involved in biological learning. One such feature is that learning occurs at the level of the individual biological neuron, whereas the ANN paradigm centralizes this process. Additionally, ANN learning algorithms don't have a notion of synaptogenesis, which has been shown to be a key component of learning in the biological brain[**?**]. Finally, there is the brain's distribution of processing among billions of individual neurons, something that is difficult to handle in fully connected paradigms of ANN [1].

2 Goals

There are essentially four goals of the OpenBrain project.

- Build a massiveley parallel Beowulf cluster of parallela computers controlled using MPI on ArchLinux.
- Create an always online, turing complete modification to the recursive neural network algorithm whose fitness is determined by the Universal Intelligence Measure described in (Legg and Veness, 2011). The algorithm must have the following constraints:
  - In the spirit of John Conway's turing complete Game of Life [6], the individual neural nodes must follow arbitrarily simple rules in a decentralized fashion.
  - *Training* is unsupervised and occurs over the lifetime of an *instance* of the open brain, such that the aforementioned governing rules are modified with respect to the fitness of the instance.
- Implement each neural node as an Erlang process distributed across the Beowulf cluster asynchronously.
- Provide always on input/output to the OpenBrain cluster in similar fashion to that done in Google DeepMind's Deep Reinforcement Learning.

3 Plans

The OpenBrain project will commense in two phases. First the algorithm must be precisely theorized and mathematical guarentees about its capabilities must be developed. Then the project will implement the algorithm through the proper hardware and software.

For the theory behind OpenBrain, we apply the neuron model proposed in McCulloch and Pitts [2] and assume that each neuron functions according Conwaynian turing complete rules; that is in particular, we apply the synaptogenesis model suggested in (Thomas et al, 2015) with parameters dictated by a universal intelligence indicator function.

For any neuron π in the OpenBrain algorithm, a list of axonally connected posterior neurons P_π is stored along with a list of proximal neighboring neurons, N_π. The neuron π is efficiently implimented as asyncrhonous thread with a message queue such that neurons connected to its anterior can activate and provide a voltage on π such that π itself will activate. Furthermore, each neuron has the capacity to signal to a particular proximal neuron in order to form a new dendritic or axonal synapse.

Under this lightweight model, we will experiment freely with different schemes of synaptogenesis and learning. Upon finding the most appropriate model, guarentees will be made on its computational capacity of the algorithm, and thenthe second phase of the project will commense.

The second phase involves implementing the algorithm on a massive parallel computing system, therefore the initial focus of this phase is to construct a 32 node Beowulf cluster of Parallela computers. The necessity for this construction is due to the large amount of CPU hours required to have the algorithm run continually, something that would not be cost effective on clusters such as Berkeley's own PSI cluster. Furthermore, the ability to control the hardware of the project provides the opportunity to maximize the performance of the algorithm with respect to the available processing power.

Once completed the child nodes of the cluster will be linked directly to a build system using hooks from Github which automatically pushes Erlang OTP code for the OpenBrain algorithm to every node. This will allow for easy prototyping and utilizes Erlang's live code hot swapping feature.

Finally much research can be done in a variety of external environments using the universal intelligence indicator and other metrics that evaluate performance. Specifically, OpenBrain will be compared to algorithms like Deep Reinforcement Learning. These applications and comparisons will be understood with more specificity as the project matures.

4 Qualifications

Coursework

1. Honors Real Analysis (MATH H104) [A] - Useful tools for providing mathematical guarentees on computability.

2. Data Structures and Algorithms (CS61B) [A] - Useful for ensuring optimal computational complexity as it pertains to the project.

Past Research

Over the past year, I've been developing an algorithmic generalization of McCulloch and Pitts feed-forward neural networks called functional neural networks. The project has provided me the education and the toolset to give actuall guarentees on a variety of different

neural network algorithms. The project received a prize from the AAAI at the Intel International Science and Engineering Fair. Attached is a copy of the paper, currently in progress, describing the project. Hopefully that may provide a sample of the quality of research I will do.

Miscelaneous

Over the past semester I've attended Peter Bartlett's Statistical Machine Learning reading group weekly.

5 Significance of the Project

The aim of this project is to reapproach the biological model of the neural network through an alternative perspective. Through the treatment of neurons as independent components, we aim to develop intelligence as an emergent phenomenon. Our method for implementing the algorithm utilizes a novel architecture through the capabilities of erlang, specifically that the implementation will be entirely asynchronous. This provides a number of benefits including the ability to use massively distributed computing and therefore easily scales.

6 Budget

The majority of the budget of this project deals directly with the construction of the Beowulf cluster.

Budget				
Item	Use	Quantity	Price	Total
Parallela	Used as the primary computation unit for the cluster. Has 18 discrete cores and low power consumption.	32	$100	$3200
8 GB Class 10 SD Card	Storage for ArchLinux on each parallela node.	32	$8.79	$281.28
Standoff M/F M3 25mm	Structureal support for nodes.	74	$0.40	$27.79
2ft Cat6 Network Ethernet Patch Cable	For connecting the nodes to a network switch.	32	$2.11	$67.60
Cisco SF200-48 Switch 10/100	Network switch for placing all nodes on a subnet.	1	$283.53	$283.53

References

[1] Nicolaos Karayiannis and Anastasios N. Venetsanopoulos *Artificial Neural Networks: Learning Algorithms, Performance Evaluation, and Applications* 2013: Springer Science Business Media.

[2] Warren S. McCulloch and Walter Pitts *A Logical Calculus of the Ideas Immanent in Nervous Activity* 1943.

[3] Frank Rosenblatt *The perceptron: a probabilistic model for information storage and organization in the brain.* 1958: Psychological review, 65(6), 386.

[4] Paul Werbos *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences.* 1974: PhD thesis, Harvard University.

[5] Monica Hoyos Flight *Synaptogenesis: Switching to learn* 2011: Nature Review Neuroscience.

[6] Martin Gardner *Mathematical Games The fantastic combinations of John Conway's new solitaire game life* 1970: Scientific American 223 pp. 120123