Andrew Simonetta

03/10/2020

CIS400M001 HW3

Particle Swam Optimization on Neural Network

The assignment was to train a Neural Network with Particle Swarm Optimization and compare the results with a Genetic Algorithm, Evolution Algorithm and the use of back propagation. I started this process on the Google cloud where I used one of their GPU. I used the neural network from the previous assignment. It has an input layer, a hidden layer, and an output layer. It functions with a binary classifier where weights and correct samples are randomized.

I implemented the Particle Swarm Optimization with 3 nodes. Each node was randomly assigned a set of weights, the weights were compared and a leader particle was found. Through 3 iterations I moved each particle, except for the lead, with movement based on the particles own findings and leader’s findings. The coefficients I used were 0.02 for the individual and 0.03 for the leader. After each iteration of the movement function I re-compared the weights of each node, determined the fitness, and assigned a new leader particle if applicable.

Just to mention a few notes about the structure of my program. I stored the particles, the weights, in a Numpy array with their loss values and when I needed to perform a function I iterated over them. I set up several helper functions to print, set, or get weights based on what I needed. The movement function was based on two factors, firstly the leader’s position. The difference between leader and the individual was found and the 0.03 coefficient was applied and it was added to the movement , since it was in the best immediate spot. Secondly, I did a quick search around the individual to determine their preferred next location. I applied this to each layer. The individual movement was an incremental change to one of the layers, a constant, to see if moving the layer in that direction was beneficial and the change was kept if it was, it did this for each layer – by a constant.

My findings were favorable and each particle improved, most of the time. Since, movement was based on two, sometimes unlinked, variables there was the possibility of traversing a less favorable spot based upon their combination. However, the overarching pattern was improvement. I was able to reach a near perfect accuracy relatively quickly. A improvement I would make if I continued work on my code would be to implement correction, the weights could leave the sigmoid range which isn’t ideal, so it would be beneficial to bound them to it in some matter. I did that in my Evo algorithm, but didn’t get around to implementing it in my PA. However, the overall idea was easily recognizable.

The major differences between Particle Swarm Optimization, Genetic Algorithm, Evolution Algorithm, and Back-propagation are pretty obvious. Each method is simply a way of searching the loss map of a particular problem based upon whatever neural net structure you use. Back-propagation is an incremental, sometimes velocity based, approach applied to one node, or population, to search the immediate area, relatively, for the optimal position. Genetic Algorithm in contrast is a stationary approach that births new weights, finds the fittest and merges their weights in new and different ways, there is no movement, but rather propagation of locations on the map catalyzed by using favorable locations genetics. Evolutionary Algorithm is similar in many regard to Genetic Algorithms, but it relies more on mutation, a sort of ‘formula’ based approach to modify the genetics of a favorable node into a more favorable node in an effort to refine a pattern to construct a proper ‘formula’ to compound this.

Particle Swarm Optimization is very different from GA and EA in the fact that it uses movement of the node to find better locations. This is similar to back-propagation, but is differentiated based upon the fact that more nodes are used and they use a movement based system in consideration of the best node and the own individual node’s findings. Depending on the parameters used, as with all algorithm really, you augment their characteristics and can create variations of the overarching idea. In essence they are related by the simple matter that they search a loss map, but are differentiated by the method they do it.