NEURAL NETWORK OPTIMIZATION USING ADAPTIVE HILL CLIMBING & SIMULATED ANNEALING

- Sridhar Natarajan

Background

I HAVE DEVISED VARIANTS OF HILL CLIMBING AND SIMULATED ANNEALING ALGORITHMS THAT FINDS BETTER NEURAL NETWORK WEIGHTS AS COMPARED TO BACK PROPAGATION AND STANDARD OPTIMIZATION ALGORITHMS FOR A LARGE DATASET. THE DOCUMENT DESCRIBES THE EXPERIMENT AND CAPTURES COMPARISON OF RESULTS.

RANDOMIZED OPTIMIZATION

Randomization algorithms compute x from possible input space of a function f(x), such that choice of x maximizes/minimizes f(x)

HILL CLIMBING

Hill climbing is an optimization technique gradually improves a solution iteratively by selecting the best neighbor based on an evaluation function until there is not a neighbor better than the current. If there is more than one best successor, a random from the set of best successors is selected. Hill climbing can get stuck at local optimums and a random restart variant helps get rid of local optimums

SIMULATED ANNEALING

Simulated annealing exploits nearly the same way as hill climbing except that it does a better job at exploring the input space. Consider the function is initially at x and jumps to xt within a neighbor space. We know HC & SA would keep xt if f(xt) improves over f(x).However if f(xt) is suboptimal than f(x), SA advocates a stochastic transition defined by a probability function ***POW (****e, (f(xt)-f(x))/T****)*** where T is a decay factor that starts with a large value and gradually decreased. The probability transition function is designed with the intent of encouraging exploration during either of the two scenarios:

1) Early phases of iteration (when T is relatively high)
2) Newfound f(xt) values are only insignificantly sub-optimal than f(x)

HILL CLIMBING	SIMULATED ANNEALING
Neighbor Function	Neighbor Function
	Initial Temperature
	Decay Rate

NEIGHBOR FUNCTION -ALGORITHM

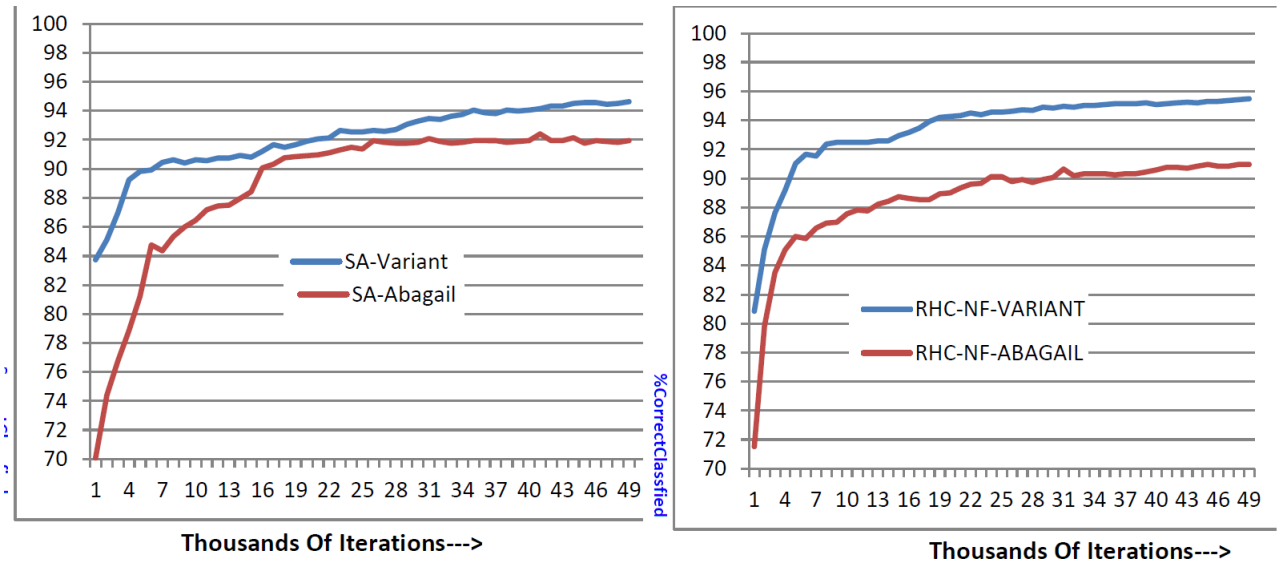
It has been observed that the choice of the neighbor function has most impact on the performance of the neural network whose weights were searched by the randomization algorithms. The neighboring function that would that at any given point maintains three values for adjusting the treading of the algorithm.

1) Neighboring function maintains three values for adjustment that is being applied to the space from where the weights are sampled. One adds a narrow sample space, one a nominal and another one a wider space to sample the weight.

2) The error values corresponding to all these deploying all three neighbor spaces are be tracked after each fixed set of iterations and the optimization algorithm's neighboring function reinforces the result into choosing the weight adjustment. If the error values corresponding to the narrower adjustments have flattened, then we apply a wider adjustment to the window in subsequent iterations and keep doubling the window adjustment until the error flattens again. Once the error flattens we are again at a decision point where we decide either to tread slowly, faster or keep the same pace.

4) The other parameter we control for treading RHC is the number of weights to which an adjustment is applied. It was also noticed that changing about 5 weights at a time to move around the neighborhood helped improve minimize the error.

Performance HILL CLIMBING (BELOW) & SIMULATED ANNEALING (TOP) ADJUSTED NN PERFORMANCE:



Although the neural network performance is not comparable with modern back propagation implementations, such an RHC/SA variant could be put to use in scenarios where gradient methods cannot be applied (when objective function is not differentiable).