Artificial Immune Systems

Lee Hall

Fri 07 Dec 2012 01:39:09 AM EST

1 Abstract

An Artificial Immune System (AIS) is a novel approach to guided classification that combines traditional genetic algorithms with a model of biological immune systems. Our approach is based on the work of von Zuben and de Castro[?], which was in turn based on the earlier work of D. Dasgupta[?]. We attempted to apply these techniques to hyperspectral image classification, by generalizing an existing classifier that recognized vectors in $\mathbb{N}^3$ to work with vectors in $\mathbb{R}^n$. This generalization revealed flaws in the underlying codebase that gave inconclusive results on the original project goals, but did provide some interesting information on the utility of various parts of the AIS algorithm.

2 Introduction

One of the more common means of building a classifier is a Genetic Algorithm (GA). A GA initially generates a random set of states. It then applies a fitness heuristic to select the most fit states. It then combines these states and applies random perturbations before repeating the process in an attempt to converge on an ideal state[?]. The AIS system is a relative of the GA, also biologically inspired, which uses simulated annealing to converge on a match for each element of the training set. The matching set, known as the Antibody set, is then produced by combining the matches with the highest fitness, and then pruning elements that are too closely related to each other.

3 ImagePimp

The starting point for our research was an existing application known as ImagePimp, which was built to do basic image manipulation and testing. There was an AINet module in this program, which would load a training set from a text file and attempt to classify the existing image based on that training set. After some work splitting the AINet code into an independent class, an attempt was made to generalize the codebase. While doing so, it became apparent that the code was extremely fragile and was implementing an algorithm much closer to a random walk than the AINet technique we desired.

4 Benchmarking Harness

When confronted by the AINet code that existed, it immediately became clear that a testing harness must be implemented as quickly as possible so that the performance of successive iterations could be measured and regressions could be easily detected. As we had been provided with several sets of classified data in text file format, scripts were written to select a randomize portion of the classified data as a training set, and to generate an unclassified copy of the original data set. A database was built which could track these benchmarks by recording the revision from the source control system as well as relevant parameters, like the size of the Antibody set, and the number of iterations allowed for convergence of the Antibody set. The benchmarking utility used 4 data sets of classified and decoded picture data: 2 ground_training sets of 3 dimensional data, 1 iris set with 4 dimensions, and the wine set with 16 dimensions. Later a 2-d set was generated based the SPIR problem[?]. The initial results were not auspicious: Table 1.

[?]

name	iterations	round	Percent Correct	Percent Stddev	trials
ground_training1	5	0.00	79.56	15.95	14
ground_training2	5	0.00	69.81	7.08	14
iris	5	0.77	4.67	11.43	14
wine	5	0.59	0.60	0.50	14

Table 1: Initial Results