Neural Network Follies

by Neil Fraser, September 1998

In the 1980s, the Pentagon wanted to harness computer technology to make their tanks harder to attack.

The Plan

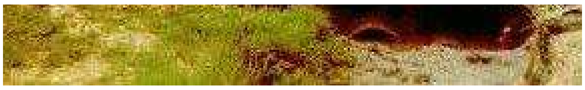
The preliminary plan was to fit each tank with a digital camera hooked up to a computer. The computer would continually scan the environment outside for possible threats (such as an enemy tank hiding behind a tree), and alert the tank crew to anything suspicious. Computers are really good at doing repetitive tasks without taking a break, but they are generally bad at interpreting images. The only possible way to solve the problem was to employ a neural network.

The Implementation

The research team went out and took 100 photographs of tanks hiding behind trees, and then took 100 photographs of trees - with no tanks. They took half the photos from each group and put them in a vault for safe-keeping, then scanned the other half into their mainframe computer. The huge neural network was fed each photo one at a time and asked if there was a tank hiding behind the trees. Of course at the beginning its answers were completely random since the network didn't know what was going on or what it was supposed to do. But each time it was fed a photo and it generated an answer, the scientists told it if it was right or wrong. If it was wrong it would randomly change the weightings in its network until it gave the correct answer. Over time it got better and better until eventually it was getting each photo correct. It could correctly determine if there was a tank hiding behind the trees in any one of the photos.

Verification

But the scientists were worried: had it actually found a way to recognize if there was a tank in the photo, or had it merely memorized which photos had tanks and which did not? This is a big problem with neural networks, after they have been trained you have no idea how they arrive at their answers, they just do. The question was did it understand the concept of tanks vs. no tanks, or had it merely memorized the answers? So the scientists took out the photos they had been keeping in the vault and fed them through the computer. The computer had never seen these photos before -- this



would be the big test. To their immense relief the neural net correctly identified each photo as either having a tank or not having one.

Independent testing

The Pentagon was very pleased with this, but a little bit suspicious. They commissioned another set of photos (half with tanks and half without) and scanned them into the computer and through the neural network. The results were completely random. For a long time nobody could figure out why. After all nobody understood how the neural had trained itself.

Grey skies for the US military

Eventually someone noticed that in the original set of 200 photos, all the images with tanks had been taken on a cloudy day while all the images without tanks had been taken on a sunny day. The neural network had been asked to separate the two groups of photos and it had chosen the most obvious way to do it - not by looking for a camouflaged tank hiding behind a tree, but merely by looking at the colour of the sky. The military was now the proud owner of a multi-million dollar mainframe computer that could tell you if it was sunny or not.

*This story might be apocryphal, but it doesn't really matter. It is a perfect illustration of the biggest problem behind neural networks. Any automatically trained net with more than a few dozen neurons is virtually impossible to analyze and understand. One can't tell if a net has memorized inputs, or is 'cheating' in some other way. A promising use for neural nets these days is to predict the stock market. Even though initial results are extremely good, investors are leery of trusting their money to a system that **nobody** understands.*

Neil Fraser: Writing: Neural Network Follies *Last modified: 23 February 2003*