Jeff Hawkins on the Limitations of Artificial Neural Networks

October 22, 2014 by Augustus Van Dusen

The media hype surrounding artificial neural networks (ANN) is becoming absurd. These articles tend to be uncritical summaries of company press releases. The most over the top articles are those derived from press releases of companies seeking funding.

ANNs are useful machine learning algorithms. They perform reasonably well in static image recognition, some types of pattern recognition, and other tasks. However, they are far too simple in how they model time, feedback, etc. Also, they are very inefficient. Companies such as Google, Baidu, and Microsoft have used enormous numbers of servers to accomplish image recognition and language translation via ANNs. Indeed, the recent surge in so-called deep ANNs is largely due to better and cheaper computing power and some clever algorithmic tricks. As a machine learning algorithm, ANNs have been very successful and will continue to be a useful tool for a growing number of problems in an ever increasing number of areas.

However, the inherent limitations of ANNs will prevent them from even approaching artificial general intelligence (AGI; see "Artificial General Intelligence Versus Narrow Artificial Intelligence (Machine Learning)") and will limit their use in machine learning. While much of the hoopla about ANNs has been about their success in image recognition, it has been realized that there are problems lurking. In "The Flaw Lurking In Every Deep Neural Net by Mike James", I linked to an article that cites a paper demonstrating two problems with ANNs.
On the NuPIC mailing list, Jeff Hawkins made some interesting comments about the limitations of ANNs.
'Thanks for finding this. It is very interesting. I haven't read the original paper yet, but I have preliminary thoughts that might be worth sharing.
The first problem they point out is that low level DL (deep learning) neurons are no better than random neurons. This sounds remarkably like our experience using an untrained spatial pooling function, what we call a "random SP". We have found that using an untrained spatial pooling function works remarkably well. So well that we often don't train the SP synapses until late in the development process. Spatial pooling is a mapping process and a random SP does the mapping just fine. The problem with untrained spatial pooling is that small changes in the input can lead to a large change in the output. This is what they are reporting in the second problem in the paper. A properly trained SP function is better than random and does not exhibit the "small change in input causes a large change in output" problem.
I don't know why the DL neurons are not learning to be better than random. I don't have a good enough insight into exactly how DL works to say. But we do understand a very similar phenomenon in HTM networks and we know that a properly trained HTM network does not have this issue.

On a more broader analysis we know that DL networks are not at all like the neural networks in biological brains. Anyone who says that ANNs and DL networks are similar to biological neural networks doesn't know much about brains. The biological and DL networks are so different that the brittleness they are seeing almost certainly does not exist in biological brains.

In a couple of recent talks I have started to point out the difference between ANNs and biological networks to drive home that HTM networks are much closer to biology.

– biological and HTM neurons have active distal dendrites, ANN neurons don't
– biological and HTM neurons have thousands of synapses, typical ANN neurons have dozens
– biological and HTM neurons have unreliable, low precision, synapses, most ANN neurons rely on synaptic weight precision
– biological and HTM neurons learn mostly by forming new synapses, ANN neurons only learn by synaptic weight modification

A typical biological neuron has 100's of proximal synapses (say 500) and these sum linearly at the soma. The proximal synapses represent only 10% of the synapses on a cell. They correspond most closely to the dozens of synapses on ANN neurons. However, there is a huge difference in how biological neurons use their proximal synapses. The proximal synapses on a biological neuron recognize dozens of unique feedforward patterns, not one. In HTM theory this is how Temporal Pooling is implemented. A cell learns to respond to dozens of unique feedforward input patterns. Temporal pooling is an absolute requirement for inference and every neuron is doing it. For a cell to recognize multiple unique patterns on its proximal dendrites requires that the patterns to be recognized are sparse. So the entire process can only be understood in the context of SDRs.

Again, I am not an expert on deep learning but I have yet to meet a DL expert who understands these concepts or the relevant biology. I am certain that HTM networks applied to applications like vision will not exhibit the problems talked about in this paper. Brains won't either.'

Note that SDR stands for sparse distributed representation and HTM stands for hierarchical temporal memory.

ANNs and Hawkins' HTM are both useful machine learning technologies that perform useful tasks, as is true of many other algorithms. None of them have reached the point to be considered universal algorithms that outperform all others on all problems. Thus, it is up to individuals to select and implement the best algorithm for a specific task.