TW414 / PR813: Lecture 10 Introduction to Artificial Neural Nets (ANNs)

This document is also available in PDF[1] format.

Purpose: Introducing basic ANNs via the concept of decision boundaries

Material: Papers by RP Lippman, Hush; Course notes by MI Jordan; *Neural Networks, a Comprehensive Foundation* by S Haykin (Macmillan).

General: We now start with another important topic for pattern modelling applications. ANNs were originally inspired by their biological counterparts, but should not be confused (idealised?) as such. They function as very simple localised processors which in combination can represent complex patterns. In this first introduction the perceptron and multi-layer perceptrons (MLPs), and their representational power, will be discussed. Many other configurations are available, some of which can be found in the quite excellent introductory articles by Lippman and Hush. We will add some others as the course progresses. Training these networks is also reserved for a later lecture.

Topics:

- Perceptron structure, activation functions and representational power.
- A single layer of perceptrons as classifiers - its limitations.
- MLPs and its capabilities (Lippman).
- Matrix formulation of the forward MLP computation (Jordan).
- Radial Basis Functions.
- Time-delay Neural Nets.
- Elman and BPTT Recursive networks.
- Hidden Control Neural Nets (HCNN).

Task (SELFSTUDY):
Although fairly old, the Lippman article is a good introduction and covers a wide range of architectures. You MUST read it thoroughly. Play around with some of the smaller configurations such as the Hopfield net associative memory. The Hush paper also provides coverage of the basic ideas, but focus on more advanced concepts. Also a must-read.

Below is a segmentation of a two-dimensional plane into 2 decision categories. Use the Jordan matrix formulation to implement this. Manually determine the number of perceptrons and their weight parameters to do this mapping. The input layer must have linear activations while the rest can be sigmoidal (initially use threshold functions). As you will realise, the weight matrices together with the activation functions define the MLP. In your implementation you can hardwire the activation functions, but supply the weight matrices as parameters to the procedure. It is also easier if you keep the DC offset weights in a separate vector. This sort of structure will facilitate the re-use of the software for later tasks.

[1] `http://www.dsp.sun.ac.za/pr813/lectures/10_ann_intro/10_ann_intro.pdf`