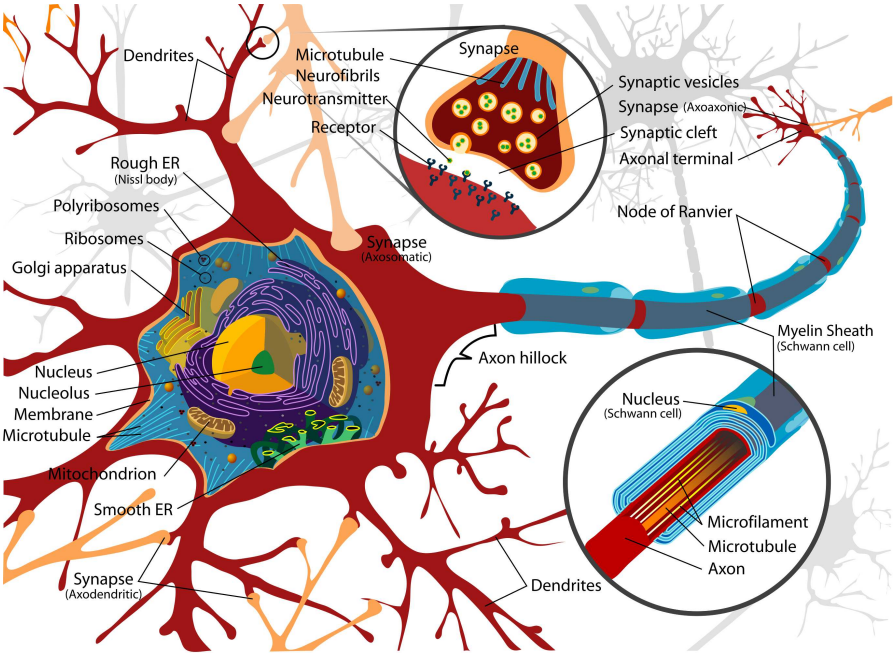
University of Cambridge
Engineering Part IB
Information Engineering Elective

Paper 8: Image Searching and Modelling Using Machine Learning

Handout 1: Introduction to Artificial Neural Networks



Roberto Cipolla and Matthew Johnson

May 2017

Notebook

You can access an interactive version of this handout at the following URL:

`https://aka.ms/ucepibieep8`

Artificial Intelligence

Complex, interconnected networks of neurons, like the human brain, are the only known way of achieving general intelligence. Artificial versions of these networks have proven helpful for more targeted AI tasks, like image understanding:

`http://captionbot.ai`

An example of this can be seen in auto-captioning systems like the one above. Not only can it recognise thousands of object categories in images, but it can perform sentiment analysis on faces and produce readable, human-like captions. Try it out now!

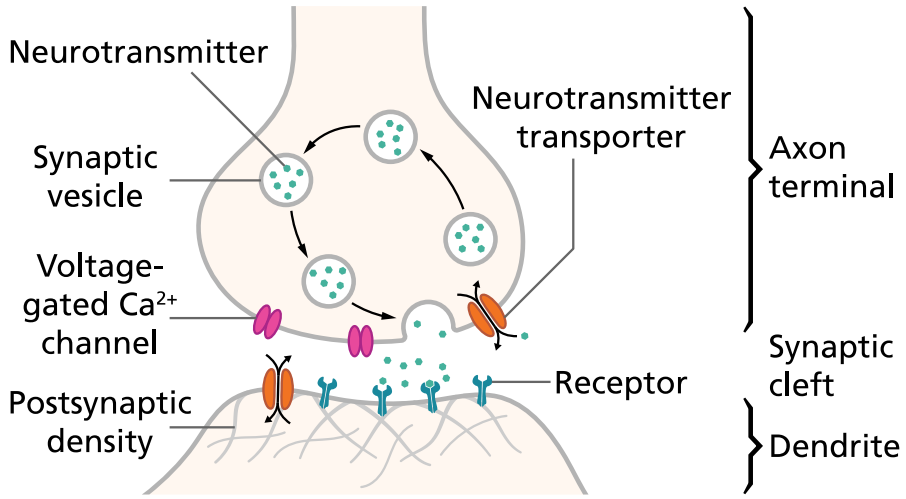
There are many publically available services like this being used to analyse images for fun or profit, including:

- `http://what-dog.net/`
- `http://how-old.net/`
- `http://clarifai.com/`

Systems like these can seem intelligent because they do things we usually associate with human intelligence, like scene understanding and object recognition. However, in reality, they are highly specialised systems trained from vast corpora of labeled data and constructed from elements that mimic in useful but incomplete ways the kinds of biological neural networks seen in animals. In the next three lectures we will explore the underlying mathematics behind these systems, analyse their structures and design principles, and study state of the art systems and techniques.

Neurons

Before we dive into the math, let us perform an (extremely brief) overview of neurons and how they work. Below is a diagram of a synapse:



The ways in which neurons interact involve complex interactions between electrical signals and neurochemistry that are very outside the scope of this course. That said, for our purposes it is important to understand three key concepts:

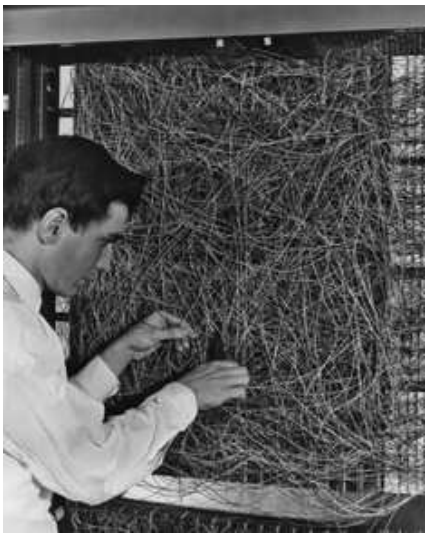
1. A neuron has both dendrites (inputs) and axons (outputs) which connect it to other neurons
2. A neuron will produce an exciting or inhibiting signal along its axons based upon activations from its dendrites in a non-linear way
3. Each dendrite contributes to whether a signal is produced in different proportions, which change over time

We will focus on these three principles and how to use them to build a simple mathematical model of a neuron.

Perceptrons

The perceptron algorithm was invented in 1957 at the Cornell Aeronautical Laboratory by Frank Rosenblatt [3]. It was designed to be implemented in hardware for the purpose of image recognition, and was the marvel of the age. The New York Times reported the perceptron to be "the embryo of an electronic computer that [the Navy] expects will be able to walk, talk, see, write, reproduce itself and be conscious of its existence." [2].

Frank Rosenblatt



The Mark 1 Perceptron

Rosenblatt based his designs upon earlier theoretical work by Warren McCulloch and Walter Pitts in 1943 on the Threshold Logic Unit [1], arguably the first artificial neuron. The idea in both cases was to create a simple mathematical model of a neuron. In the case of the perceptron, this model was then expressed in hardware. The entire machine represented a single neuron in this sense, with 400 inputs from a 20x20 image sensor and a binary output. It was later shown that its capabilities were fairly limited, restricted mostly to linearly separable classification problems.

Perceptrons, cont.

The math behind perceptrons is fairly straightforward. Given an input $\mathbf{x}$, the output of a perceptron is calculated as:

$$y(\mathbf{x}) = f(\mathbf{w}^{\mathrm{T}}\mathbf{x}) \tag{1}$$

where $\mathbf{w}$ are the weights on the input vector $\mathbf{x}$ and f is a step function of the form:

$$f(x) = \begin{cases} +1 & x \geq 0 \\ -1 & x < 0 \end{cases} \tag{2}$$

The fundamental problem at hand is how to learn the values of $\mathbf{w}$. For this we use the *perceptron criterion*:

$$E_{\mathrm{P}}(\mathbf{w}) = -\sum_{n \in \mathcal{M}} \mathbf{w}^{\mathrm{T}}\mathbf{x}_n t_n \tag{3}$$

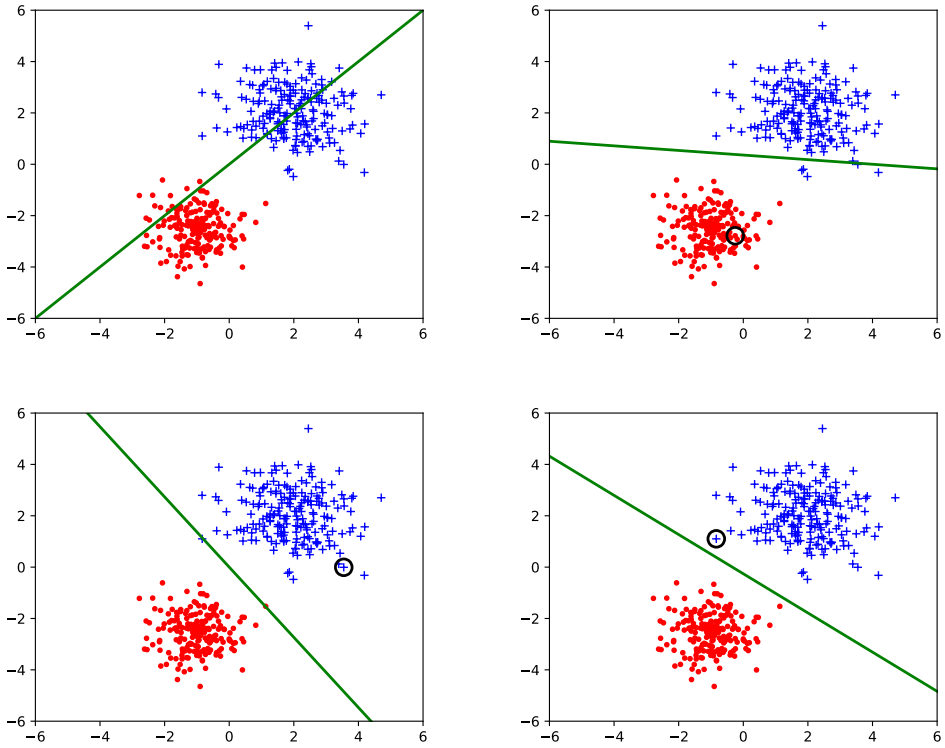
where $t_n \in \{-1, +1\}$ for negative and positive examples and $\mathcal{M}$ is the set of misclassified examples. We can use the gradient of this function to update the weights on a per-example basis:

$$\mathbf{w}^{\tau+1} = \mathbf{w}^{\tau} - \eta \nabla E_{\mathrm{P}}(\mathbf{w}) = \mathbf{w}^{\tau} + \eta \mathbf{x}_n t_n \tag{4}$$

where η is the learning rate parameter and τ is an integer that indexes the steps of the algorithm.

Training a Perceptron

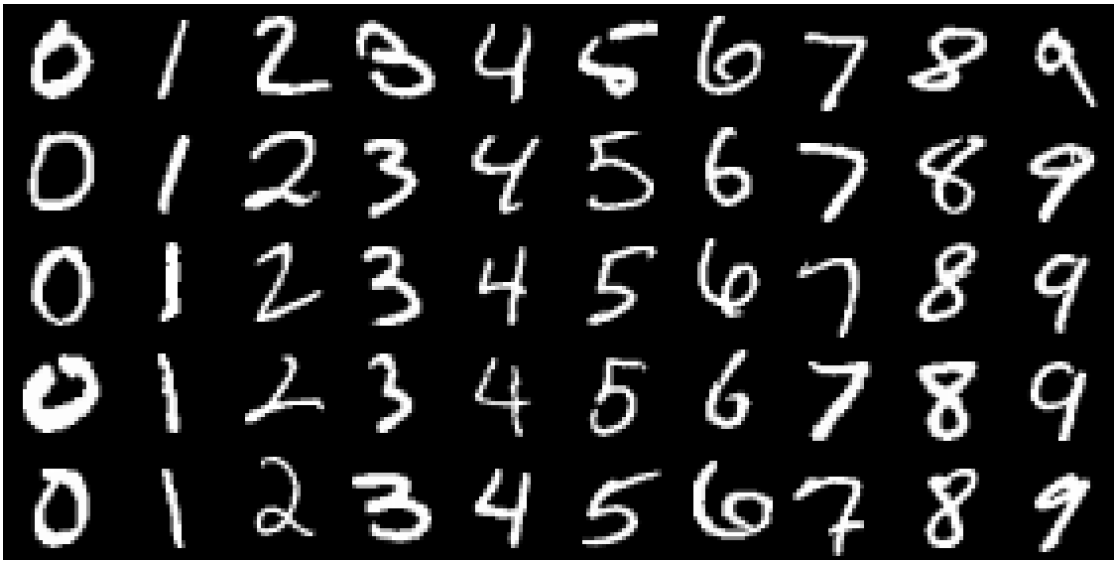
Below is a sample problem. There are two classes, represented here by two sets of points on a plane. The perceptron's decision boundary is represented by the green line.



The top left corner is the initial state. First, a red circle is presented, and the perceptron adapts its boundary. This one example solves most of this problem, and it is only when the circled blue cross in the bottom left is presented that the boundary changes again. Finally, the algorithm finds one more error in the bottom right and uses it to correct the perceptron, resulting in a perfect linear boundary between the two classes.

Recognising Digits

An important task in the history of artificial neural networks is the recognition of handwritten digits, driven by a dataset known as MNIST.

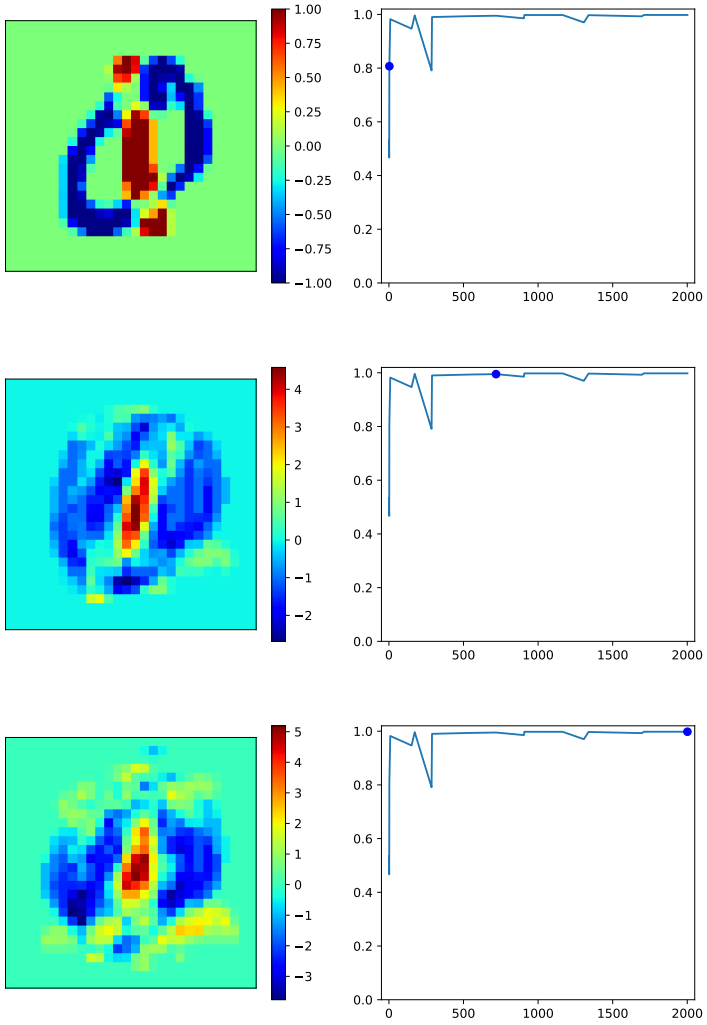


The images above were gathered from employees of the US Census Bureau and Secondary School students, consisting of approximately 250 individuals. There are 60,000 training images and 10,000 test images (sampled in a disjoint manner from the two sample groups). The images themselves are 28×28 unregistered grayscale images. Research has been performed on the dataset since the 1980s and has continued to the present day where it is often used in tutorials for neural net software frameworks.

In Rosenblatt's original work, he trained perceptrons to tell the difference between two different images of numbers. Using this new dataset, we can replicate those experiments, though we will do it entirely in software of course.

1 vs 0

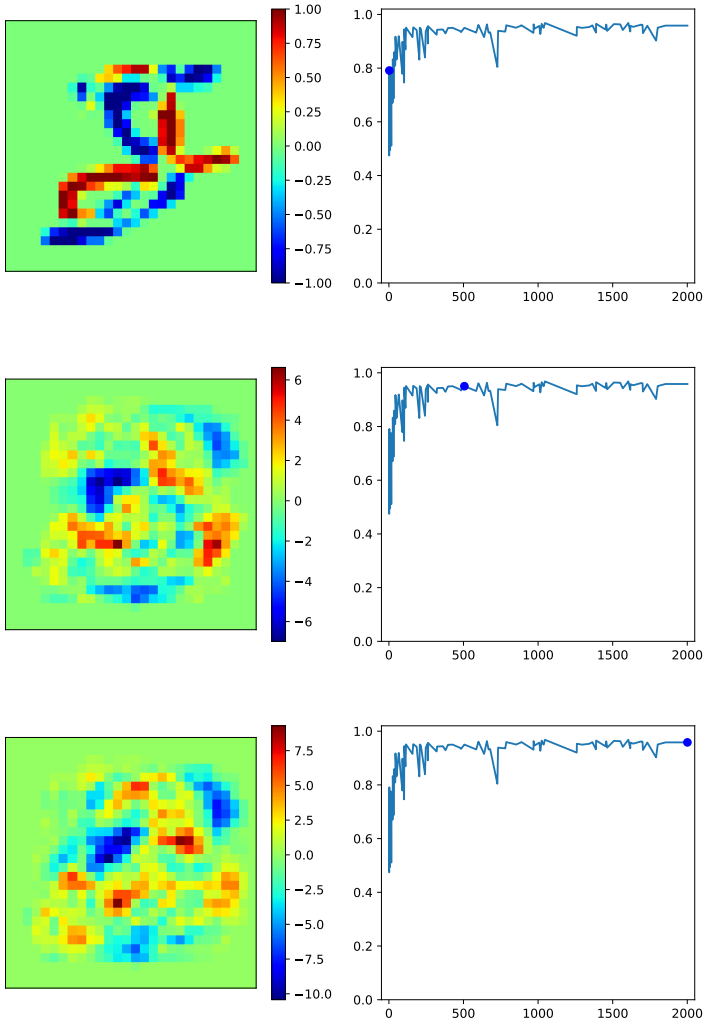
First, we look at the result of training a perceptron to distinguish between hand-written 1s and 0s from the MNIST dataset. Below are snapshots from the 2nd, 717th and 2000th image presented to the perceptron:



As you can see, the perceptron is quite effective at solving this problem, achieving a final accuracy of 0.998. It clearly learns that the positive class (1) has a vertical bit in the center, while the negative class (0) is shaped like a ring.

2 vs 5

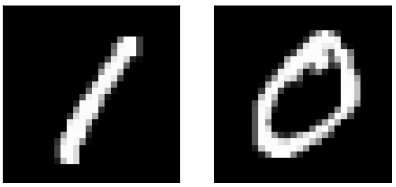
Now we will examine what happens when we train a perceptron to distinguish two quite similar numerals, 2 and 5. Below are snapshots from the 2nd, 505th and 2000th image presented to the perceptron:

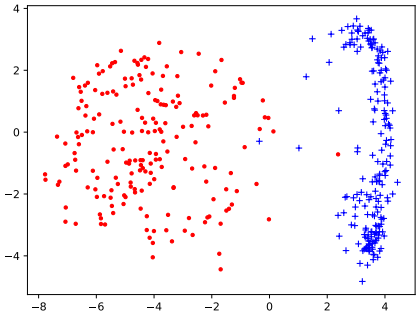


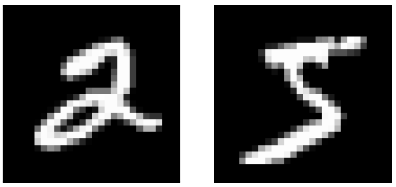
The perceptron seems to have more difficulty with this task, achieving a lower accuracy of 0.958 and seeming to plateau. Indeed, if this perceptron is presented with more examples it will never converge and eventually causes a numerical overflow.

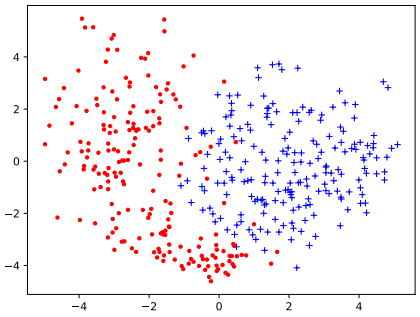
Visualisation

Why does the perceptron solve one problem almost perfectly, and seem to stumble on the second? Below are some helpful visualizations of a low-dimensional embedding of the different pairs of image classes.









Each point represents a different digit image. Note how, even in this low-dimensional space, the 1s and 0s are linearly separable? Contrast this with the 2s and 5s, which are all mixed up. There is not a line that separates these two classes, and that is why the perceptron will never perfectly achieve the task of distinguishing them.

In the next lecture, we will examine how to combine perceptrons in different ways to overcome these limitations.

References

[1] Warren S. McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, 5:115–133, 1943.

[2] Mikel Olazaran. A sociological study of the official history of the perceptrons controversy. *Social Studies of Science*, 26(3):611–659, 1996.

[3] Frank Rosenblatt. The Perceptron - a perceiving and recognizing automaton. Technical Report 85-460-1, Cornell Aeronautical Laboratory, 1957.