works (BNN)[4] is still the subject of active research, but some parts of the brain have been mapped, and it seems that neurons are often organized in consecutive layers, as shown in Figure 10-2.

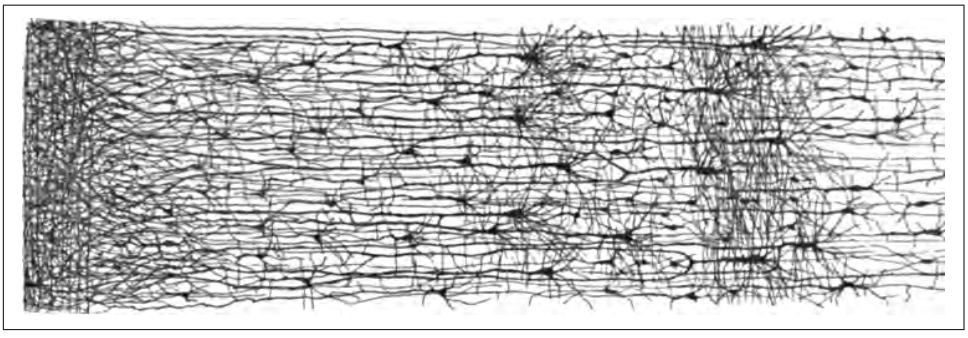

Figure 10-2. Multiple layers in a biological neural network (human cortex)[5]

Logical Computations with Neurons

Warren McCulloch and Walter Pitts proposed a very simple model of the biological neuron, which later became known as an *artificial neuron*: it has one or more binary (on/off) inputs and one binary output. The artificial neuron simply activates its output when more than a certain number of its inputs are active. McCulloch and Pitts showed that even with such a simplified model it is possible to build a network of artificial neurons that computes any logical proposition you want. For example, let's build a few ANNs that perform various logical computations (see Figure 10-3), assuming that a neuron is activated when at least two of its inputs are active.

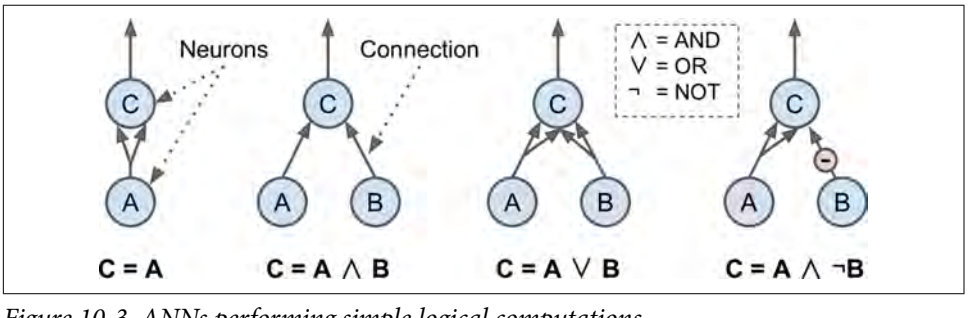

Figure 10-3. ANNs performing simple logical computations

4 In the context of Machine Learning, the phrase "neural networks" generally refers to ANNs, not BNNs.

5 Drawing of a cortical lamination by S. Ramon y Cajal (public domain). Reproduced from *https://en.wikipedia.org/wiki/Cerebral_cortex*.

- The first network on the left is simply the identity function: if neuron A is activated, then neuron C gets activated as well (since it receives two input signals from neuron A), but if neuron A is off, then neuron C is off as well.
- The second network performs a logical AND: neuron C is activated only when both neurons A and B are activated (a single input signal is not enough to activate neuron C).
- The third network performs a logical OR: neuron C gets activated if either neuron A or neuron B is activated (or both).
- Finally, if we suppose that an input connection can inhibit the neuron's activity (which is the case with biological neurons), then the fourth network computes a slightly more complex logical proposition: neuron C is activated only if neuron A is active and if neuron B is off. If neuron A is active all the time, then you get a logical NOT: neuron C is active when neuron B is off, and vice versa.

You can easily imagine how these networks can be combined to compute complex logical expressions (see the exercises at the end of the chapter).

The Perceptron

The *Perceptron* is one of the simplest ANN architectures, invented in 1957 by Frank Rosenblatt. It is based on a slightly different artificial neuron (see Figure 10-4) called a *linear threshold unit* (LTU): the inputs and output are now numbers (instead of binary on/off values) and each input connection is associated with a weight. The LTU computes a weighted sum of its inputs ($z = w_1 x_1 + w_2 x_2 + \cdots + w_n x_n = \mathbf{w}^T \cdot \mathbf{x}$), then applies a *step function* to that sum and outputs the result: $h_{\mathbf{w}}(\mathbf{x}) = \text{step}(z) = \text{step}(\mathbf{w}^T \cdot \mathbf{x})$.

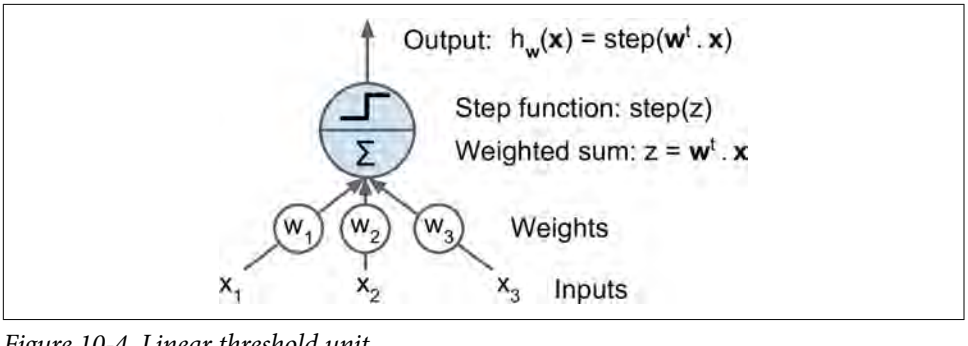

Figure 10-4. Linear threshold unit

The most common step function used in Perceptrons is the *Heaviside step function* (see Equation 10-1). Sometimes the sign function is used instead.