1
Artificial Neural Networks

There is nothing either good or bad but thinking makes it so.
—Shakespeare, *Hamlet*, II, ii

Since man's earliest efforts to build an electronic calculating machine, scientists and engineers have dreamed of constructing the ultimate artificial brain. Though we may never reach this goal, the first successful attempt to create a computer algorithm that would mimic, albeit in a much simplified way, the brain's remarkably complicated structure and function represented a significant stride forward. These algorithms, known as artificial neural nets, are defined as an interconnected group of information processing units whose functionality is roughly based on the living neuron. As these units "learn" or process information by adapting to a set of training patterns, it is reflected in the strength of their connections.

Neural nets represent a different paradigm for computing than that of conventional digital computers, because their architecture closely parallels that of the brain. (Traditional computers, based on von Neumann's design, were inspired by a model of brain function by incorporating concepts such as input, output, and memory, but reflect this only abstractly in their architecture.) Neural nets are useful for problems where we can't find an algorithmic solution, but can find lots of examples of the behavior we're looking for, or where we need to identify the solution's structure from existing data. In other words, they don't need to be programmed to solve a specific problem; they "learn" by example. They have their roots in a pioneering 1943 paper written by mathematician Walter Pitts and psychiatrist Warren McCullough, "A Logical Calculus of the Ideas Immanent in Nervous Activity." It was the first time anyone

had tried to describe the idealized behavior of the brain's network of nerve cells (neurons)—a poorly understood phenomenon at that time—in the language of mathematics and logic.

A Logical Calculus for the Brain

Though the Pitts-McCullough theory had its shortcomings—many physiologists were not happy with its treatment of the neuron as a black box that followed certain mathematical rules for input and output without taking actual physiology into account—many of their ideas were revolutionary and still survive today. They were the first to bring mathematical uniformity, based on logical axioms, to the idea of information processing in the brain. Within this framework they described a network of neurons that cooperated to sense, learn, and store information, in addition to other information processing tasks. They originated the highly sophisticated way of conceptualizing a neuron as an element that sums the electrical signals from many incoming neurons, in addition to the notion that the strength of the synaptic connection between neurons acts as a weighting function whose value determines whether the outgoing signal will excite or inhibit an outgoing electrical nerve impulse. The two researchers also originated the idea that the weighted sum of nerve signals coming into a synapse had a threshold value. According to this neural calculus, if the sum exceeded this value, the outgoing signal would be a one; if not, it would be a zero. This demonstration of the digital nature of neural behavior would come to be a key concept in the theory of artificial neural nets.

The work of psychologist Donald Hebbs also helped shape the field of artificial neural nets. His 1949 book *The Organization of Behavior* put forth the idea that the more active two connected neurons were, the stronger their synaptic connection would become. By extension, the greater the degree of electrical activity throughout a given neural pathway, the more its synaptic connections will be reinforced; this effect, on another level of abstraction, equates to "learning." In other words, according to Hebbs, when we learn something—a child learning to write the letters of an alphabet by repeated practice, for example—we are strengthening the connections in the brain's neural pathways that underlie the behavior.

A New Field Evolves

Artificial neural nets were the first computer algorithms that attempted to model not only the brain's organization, but also its ability to actually learn, based on physiological changes in the organization of neural pathways. The way one actually goes about training a neural net to perform a task is complex; however, it's a vastly simpler process—acting on a vastly simpler network structure—when compared to the biochemistry of learning in a living brain, even as we learn something as rudimentary as a one-syllable sound.

Since its origin in the 1950s, based on Pitts and McCullough's work, the field of artificial neural nets has continued to develop by drawing conceptually from advances in modern neurophysiology, as well as by repeated application at the hands of researchers. For example, a special class of artificial neurons called "perceptrons" were created in the late 1950s. The perceptron was more elaborate and more akin to biological reality than the more schematic, mathematically based Pitts-McCullough neuron, and was defined as a single layer of information processing units that transmitted signals and adapted its interconnecting weights accordingly. In the late 1960s, however, further analysis revealed that the perceptron was unable to carry out certain logic functions involved in more sophisticated learning algorithms. Though these problems were eventually solved, this failing of the perceptron slowed growth in the field and brought it to an eventual intellectual impasse, causing interest in artificial nets as a model for human intelligence to dwindle.

In the early 1980s, however, the field experienced a rebirth, largely prompted by the discovery of "recurrent networks" by Caltech physicist John Hopfield. These are networks where information flows from a connection node back to itself via other nodes, providing all the neurons in the network with complete connectivity and greater resemblance to the biological brain. (In other words, they incorporated self-feedback loops.) This development added greatly to the range of problems neural nets were capable of addressing. Hopfield was also responsible for introducing the idea of "hidden" layers of neurons between input and output layers (figure 1.1). These layers are not connected to the outside and can recode, or provide a representation for the input units. They are more

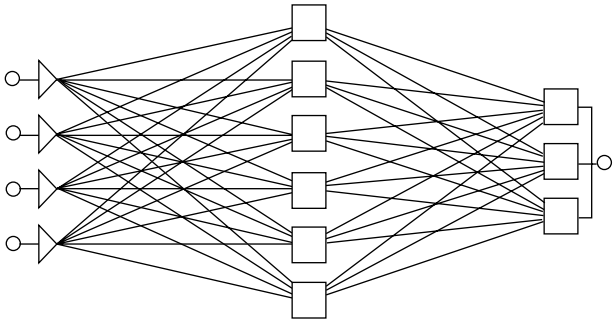

Figure 1.1
Artificial neural nets usually have one or more "hidden" layers between input and output layers.

powerful than single-layer networks, in that they can learn a much wider range of behaviors.

The next big development came in the 1980s with the arrival of the "backpropagation" algorithm. This was a procedure for training neural nets to learn from test cases or "training sets." These are presented to the net one at a time, and the errors between the actual and desired behavior of the network are propagated backward to the hidden layers, enabling them to adjust the strength of their connections accordingly. This method is then iterated to reduce the error to an acceptable one. These two advances were probably responsible for a large resurgence of interest in the field in the late 1980s, and though artificial neural nets never became the much hoped-for means for creating an electronic replica of the human brain, they did give rise to a sophisticated technology that is still widely employed today in many industrial, research, and defense applications, particularly for pattern recognition.

How Artificial Neural Nets Work

The mammalian brain is made up of a huge network of nerve cells or neurons that are specialized to carry messages in the form of electrochemical signals. In humans, the brain has more than 100 billion neurons that communicate with each other via a massive web of interconnections. These interconnections consist of nerves called dendrites, which carry input into the neuron, whereas other nerves, called axons, are its output

channels. At the connection points between the dendrites and the axons—the synapses—the electrical impulses flowing down the axon get transformed into biochemical signals, cross the synaptic gap, and are then re-transformed into electrical signals that travel up the dendrite to the next neuron. The electrical impulse passing though the dendrite is either "excitatory" (promoting action) or "inhibitory" (inhibiting action) in nature. If the difference between the sum of all excitatory and inhibitory impulses reaching the neuron exceeds a given threshold, the neuron will fire an electrical pulse. This pulse, in turn, is itself inhibitory or excitatory in nature (figure 1.2). This represents the *all-or-none* response of the neuron.

Artificial neural net algorithms are based on a highly simplified model of the brain's elaborate network of neural connections. Artificial neural nets have input channels that represent the dendrites, and output channels that mimic the axons. The synapse is modeled by an adjustable weight, located at the juncture between incoming and outgoing channels. A one or zero represents the corresponding excitatory or inhibitory signal that flows out from each connecting point (figure 1.3).

Within the artificial neural net, each connection weight modifies the incoming signal before sending it on by assigning it an appropriate weighted value. Much like what happens in the living brain, all the weighted input signals in the network that flow into that particular

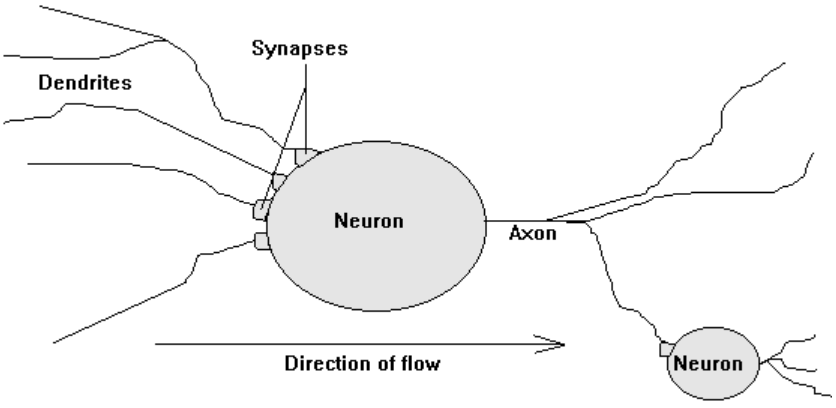

Figure 1.2
Sketch of a living neuron, showing dendrites, axons, and synapses.

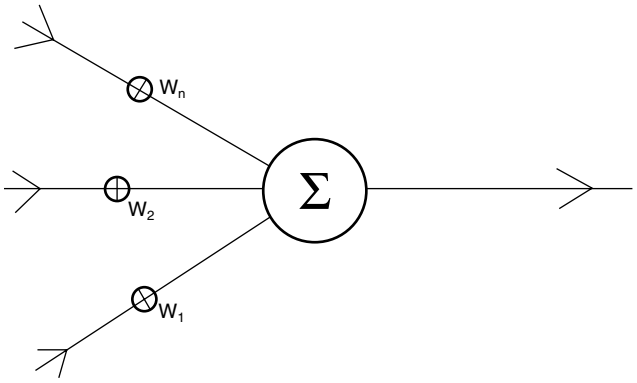

Figure 1.3
Simplified artificial neuron. Each input is multiplied by a weighting factor (w_1, w_2 . . . w_n), before flowing into the "synapse," where they are summed.

synapse are added together to form a total input signal, which is routed through something called an "input-output function." This acts on the signal to form the final output signal. The weights and the input-output function are what ultimately determine the behavior of the network, and can be adjusted (figure 1.4).

In order for the artificial neural net to carry out a useful task, one must connect the neurons in a particular configuration, set the weights, and choose the input-output functions. The simplest artificial neural net would consist of a layer of input units connected to a single middle or "hidden" layer, which is linked to a layer of output units. To initialize the artificial neural net, whatever raw data is needed to perform the task is first fed into the input units. The resulting signal received by a neuron in the hidden layer depends on how the incoming raw data is weighted, and how it is modified by the input-out function. In the same way, the signal flowing out of the hidden layer goes through a similar process of weighting and modification before going on to the subsequent level.

What makes artificial neural net algorithms so valuable is that they can be taught to perform a particular task, such as recognizing patterns inherent in an incoming data set. A concrete example may help demystify the process by which artificial neural nets learn. Suppose we want to train the network to recognize handwritten letters on a display screen (as many credit card machines do today with the card owner signature).

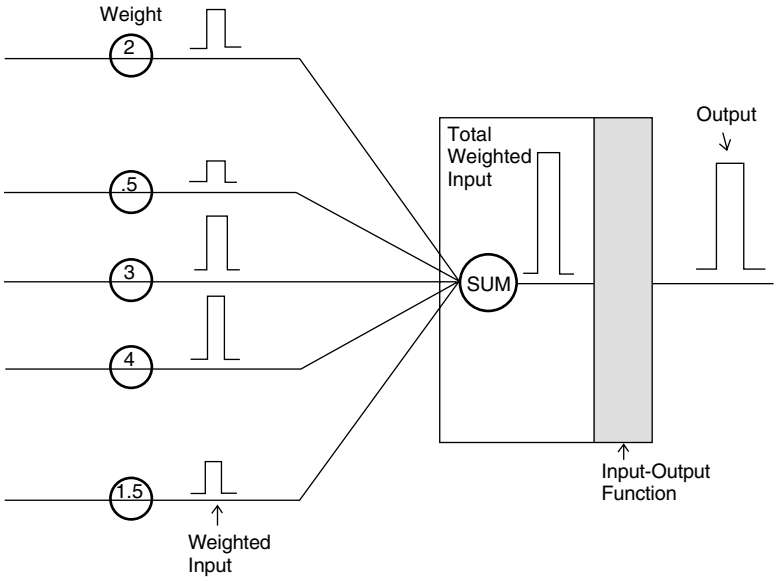

Figure 1.4
Detailed artificial neuron. All the weighted inputs are summed to form a total weighted input, which then passes through a given input-output function for computation of the final output signal.

The artificial neural net would therefore need as many input units as the number of pixels making up the screen, and twenty-six output units for each letter of the alphabet (with any number of units in the hidden layer in between).

To train this network to recognize letters, we first present it with the image of a handwritten letter, for example, "A," and compare the output this input signal produces with the desired output. (The learning task is incremental, as the network gradually learns the desired task). We then calculate the discrepancy or error between the initial output and the one we ultimately want. The error is defined as the square root of the difference between the two outputs. Then we tweak the values of the weights a little in an attempt to better approximate the desired output, continuing this iterative process until the actual output comes closer and closer and finally matches the desired output.

The difficult part, of course, is knowing how to modify the weights to increasingly reduce the error between actual and correct values.

However, in practice, the weight is adjusted by an amount proportional to the rate of change in the error, a quantity that can be calculated as the weight is changed. This quantity is called "the error derivative of the weight," or EW. It is often very hard to calculate. In 1974, Paul Werbos then a doctoral student at Harvard, invented a better way to calculate the EW, a method called backpropagation.

To understand how backpropagation works, first assume all the input-output functions are linear. To find the EW, we must first find the rate of change in the error as a particular unit's signal is changed, called the "error derivative," or EA. For an output unit in the network, this quantity is the difference between the actual output and the desired output. In backpropagation, we start backward from the output layer, computing all the EAs for the hidden layers and input layers. To find the corresponding EW for each weight in each layer, we simply multiply the EA by the signal that enters that weight. When all the weights have been adjusted by the right amount, we can input the same raw data into the network, and the actual output will match up with the desired output. In the handwriting recognition case, we can feed into the network the corresponding impulses from the pixels on the display screen as someone writes an A, and the handwriting recognition device will register "A."

Training algorithms for artificial neural nets come in many varieties, the two most common being *supervised* learning and *unsupervised* learning. In the former, an outside computer program monitors the learning process just as a teacher would do for a student; in unsupervised learning, the network is only presented input data, and the system adjusts its own weights without the benefit of knowing the relationship between the input and final output. To reach a solution, the system groups the input data into special classes, and is ultimately able to obtain a single output correlated with each group. Certain artificial neural net algorithms used to recognize spoken speech patterns are unsupervised, in the sense that they place spoken words into different phonetic classes.

Why Artificial Neural Nets Are Useful

Artificial neural nets have been successfully applied to a large number of problems, which usually fall into one of three classes: recognizing something, inferring something, or putting things into classes. Pattern recog-

nition algorithms are probably the most common. Among these are the automated recognition of handwritten text, spoken words, facial/fingerprint identification, and automatic moving target identification against a static background (for example, the ability to differentiate the image of a moving tank from the road).

Speech production is an example of a classification algorithm. Connected to a speech synthesizer, the artificial neural net is able to classify different sounds, such as vowels, consonants, and even those that separate one word from another—as an infant does when learning to speak—developing increasingly finer-grained classes of sounds until they represent actual intelligible speech. Artificial neural nets are also used for industrial control systems, such as power plants or chemical factories. The network is fed data that represents the system's optimal state of functioning (for example, the ideal temperature, pressure, and vacuum conditions) and continually monitors the system for any deviation from these values. Other applications include artificial neural net software to predict stock market trends (a pattern recognition algorithm), or to process signals while canceling out noise, echoes, and other unwanted parts of the input signal.

Artificial Neural Nets and Digital Computers

The way an artificial neural net processes information is fundamentally different from the way digital desktop computers do—although the latter can be modified to run artificial neural nets. Conventional digital computers are traditionally known as "von Neumann Machines" because they are based on von Neuman's original designs. They essentially work by deductive reasoning. This method is optimal for solving problems whose solutions can be reached by following a formalized, linear, finite series of instructions (algorithm) that the computer's central processing unit (CPU) executes. Computers must be programmed *a priori* with the exact series of steps needed to carry out the algorithm. What's more, the data fed into the program must be precise, containing no ambiguities or errors. Conventional computers are amazingly adept at carrying out what they've been programmed to do, including executing extremely complicated mathematics. They are also remarkably fast and precise. Some digital supercomputers can perform more than a trillion

operations per second and are thousands of times faster than a desktop computer.

However, traditional digital computers can only solve problems we already know and understand how to solve. They're ineffective if we're not sure what kind of problem we want to solve, know an algorithm for doing it, or if the data we have to work with is vague. But if we can point to a number of examples of the kind of solution we require, or if we simply want to find a pattern in a mass of disorganized data, artificial neural nets are the best method.

In contrast to digital computers, artificial neural nets work by inductive reasoning. Give them input data and the desired solution, and the network itself constructs the proper weightings for getting from one to the other. This is what is meant by saying that the artificial neural net is "trained" from experience—the initial network is built and then presented with many examples of the desired type of historical cause and effect events. The artificial neural net then iteratively shapes itself to build an internal representation of the governing rules at play.

Once the network is trained, it can be fed raw input data and produce the desired solution on its own—analogous to the way the brain functions in the learning process. Unlike the digital computer, where computation is centralized, serial, and synchronized, in artificial neural nets, computation is collective, parallel, and unsynchronized. They tend to be much slower at this process than digital computers—artificial neural net operations are measured in thousandths of a second, whereas digital computers can function at up to teraops, or 10^{12} operations per second rates at present.

Artificial Neural Nets and Artificial Intelligence

Though the initial impetus for developing artificial neural nets may have been a desire to create an artificial brain, in the years since Pitts and McCullough's work, research in the general area of machine learning has grown so specialized and so diverse that some of the algorithms bear little resemblance to others. Such is the case of artificial neural nets and artificial intelligence (AI), although ultimately each represents a different approach to the long-standing quest to make computers more and more humanlike in their abilities.

Artificial intelligence, like artificial neural nets, consists of computer algorithms that mimic human intelligence. They are typically used to carry out tasks such as learning, game playing, natural language processing, and computer vision. Generally speaking, artificial intelligence differs from artificial neural nets in the level of human intervention it requires. With an AI algorithm, all the information needed for a solution must preprogrammed into a database, whereas artificial neural nets learn on their own. AI is based on the principles of deductive reasoning, whereas neural nets are inductive. This means that with AI, each new situation the system encounters may require another programmed rule. For example, when AI is used to program the behavior of a robot, all the desired behavior patterns must be worked out and programmed a priori—the robot can't adapt its behavior to changes in the environment. Consequently, AI programs can become quite large and unweildy in their attempt to address a wide range of different situations.

Artificial neural nets, on the other hand, automatically construct associations or relationships between parts of the network according to the results of known situations, adjusting to each new situation and eventually generalizing their behavior by correctly guessing the output for inputs never seen before. The disadvantage of artificial neural nets, however, is that they cannot be programmed to do a specific task, like adding numbers. The sets of examples or "training sets" of data the network must be fed in order to bring it closer to the desired solution must be chosen very carefully; otherwise, valuable time is wasted—or worse, the network doesn't do what it is supposed to do.

Popular culture has conditioned us to expect the future to be populated with robots containing computer programs that will make them look and act like humans. Although this is a quixotic goal that we may never reach—and some would question whether or not it's even desirable—as more of the brain's remarkable complexity is deciphered and understood, it will likely inspire many new technological ideas, equally as impressive as these.