Artificial Neural Networks Technology

2.5 Training an Artificial Neural Network

Once a network has been structured for a particular application, that network is ready to be trained. To start this process the initial weights are chosen randomly. Then, the training, or learning, begins.

There are two approaches to training - supervised and unsupervised. Supervised training involves a mechanism of providing the network with the desired output either by manually "grading" the network's performance or by providing the desired outputs with the inputs. Unsupervised training is where the network has to make sense of the inputs without outside help.

The vast bulk of networks utilize supervised training. Unsupervised training is used to perform some initial characterization on inputs. However, in the full blown sense of being truly self learning, it is still just a shining promise that is not fully understood, does not completely work, and thus is relegated to the lab.

2.5.1 Supervised Training.

In supervised training, both the inputs and the outputs are provided. The network then processes the inputs and compares its resulting outputs against the desired outputs. Errors are then propagated back through the system, causing the system to adjust the weights which control the network. This process occurs over and over as the weights are continually tweaked. The set of data which enables the training is called the "training set." During the training of a network the same set of data is processed many times as the connection weights are ever refined.

The current commercial network development packages provide tools to monitor how well an artificial neural network is converging on the ability to predict the right answer. These tools allow the training process to go on for days, stopping only when the system reaches some statistically desired point, or accuracy. However, some networks never learn. This could be because the input data does not contain the specific information from which the desired output is derived. Networks also don't converge if there is not enough data to enable complete learning. Ideally, there should be enough data so that part of the data can be held back as a test. Many layered networks with multiple nodes are capable of memorizing data. To monitor the network to determine if the system is simply memorizing its data in some nonsignificant way, supervised training needs to hold back a set of data to be used to test the system after it has undergone its training. (Note: memorization is avoided by not having too many processing elements.)

If a network simply can't solve the problem, the designer then has to review the input and outputs, the number of layers, the number of elements per layer, the connections between the layers, the summation, transfer, and training functions, and even the initial weights themselves. Those changes required to create a successful network constitute a process wherein the "art" of neural networking occurs.

Another part of the designer's creativity governs the rules of training. There are many laws (algorithms) used to implement the adaptive feedback required to adjust the weights during training. The most common technique is backward-error propagation, more commonly known as back-propagation. These various learning techniques are explored in greater depth later in this report.

Yet, training is not just a technique. It involves a "feel," and conscious analysis, to insure that the network is not overtrained. Initially, an artificial neural network configures itself with the general statistical trends of the data. Later, it continues to "learn" about other aspects of the data which may be spurious from a general viewpoint.

When finally the system has been correctly trained, and no further learning is needed, the weights can, if desired, be "frozen." In some systems this finalized network is then turned into hardware so that it can be fast. Other systems don't lock themselves in but continue to learn while in production use.

2.5.2 Unsupervised, or Adaptive Training.

The other type of training is called unsupervised training. In unsupervised training, the network is provided with inputs but not with desired outputs. The system itself must then decide what features it will use to group the input data. This is often referred to as self-organization or adaption.

At the present time, unsupervised learning is not well understood. This adaption to the environment is the promise which would enable science fiction types of robots to continually learn on their own as they encounter new situations and new environments. Life is filled with situations where exact training sets do not exist. Some of these situations involve military action where new combat techniques and new weapons might be encountered. Because of this unexpected aspect to life and the human desire to be prepared, there continues to be research into, and hope for, this field. Yet, at the present time, the vast bulk of neural network work is in systems with supervised learning. Supervised learning is achieving results.

One of the leading researchers into unsupervised learning is Tuevo Kohonen, an electrical engineer at the Helsinki University of Technology. He has developed a self-organizing network, sometimes called an auto-associator, that learns without the benefit of knowing the right answer. It is an unusual looking network in that it contains one single layer with many connections. The weights for those connections have to be initialized and the inputs have to be normalized. The neurons are set up to compete in a winner-take-all fashion.

Kohonen continues his research into networks that are structured differently than standard, feedforward, back-propagation approaches. Kohonen's work deals with the grouping of neurons into fields. Neurons within a field are "topologically ordered." Topology is a branch of mathematics that studies how to map from one space to another without changing the geometric configuration. The three-dimensional groupings often found in mammalian brains are an example of topological ordering.

Kohonen has pointed out that the lack of topology in neural network models make today's neural networks just simple abstractions of the real neural networks within the brain. As this research continues, more powerful self learning networks may become possible. But currently, this field remains one that is still in the laboratory.

2.6 How Neural Networks Differ from Traditional Computing and Expert Systems

Neural networks offer a different way to analyze data, and to recognize patterns within that data, than traditional computing methods. However, they are not a solution for all computing problems. Traditional computing methods work well for problems that can be well characterized. Balancing checkbooks, keeping ledgers, and keeping tabs of inventory are well defined and do not require the special characteristics of neural networks. Table 2.6.1 identifies the basic differences between the two computing approaches.

Traditional computers are ideal for many applications. They can process data, track inventories, network results, and protect equipment. These applications do not need the special characteristics of neural networks.

Expert systems are an extension of traditional computing and are sometimes called the fifth generation of computing. (First generation computing used switches and wires. The second generation occurred because of the development of the transistor. The third generation involved solid-state technology, the use of integrated circuits, and higher level languages like COBOL, Fortran, and "C". End user tools, "code generators," are known as the fourth generation.) The fifth generation involves artificial intelligence.

CHARACTERISTICS	**TRADITIONAL COMPUTING (including Expert Systems)**	**ARTIFICIAL NEURAL NETWORKS**
Processing style Functions	Sequential Logically (left brained) via Rules Concepts Calculations	Parallel Gestault (right brained) via Images Pictures Controls
Learning Method Applications	by rules (didactically) Accounting word processing math inventory digital communications	by example (Socratically) Sensor processing speech recognition pattern recognition text recognition

Table 2.6.1 Comparison of Computing Approaches.

Typically, an expert system consists of two parts, an inference engine and a knowledge base. The inference engine is generic. It handles the user interface, external files, program access, and scheduling. The knowledge base contains the information that is specific to a particular problem. This knowledge base allows an expert to define the rules which govern a process. This expert does not have to understand traditional programming. That person simply has to understand both what he wants a computer to do and how the mechanism of the expert system shell works. It is this shell, part of the inference engine, that actually tells the computer how to implement the expert's desires. This implementation occurs by the expert system generating the computer's programming itself, it does that through "programming" of its own. This programming is needed to establish the rules for a particular application. This method of establishing rules is also complex and does require a detail oriented person.

Efforts to make expert systems general have run into a number of problems. As the complexity of the system increases, the system simply demands too much computing resources and becomes too slow. Expert systems have been found to be feasible only when narrowly confined.

Artificial neural networks offer a completely different approach to problem solving and they are sometimes called the sixth generation of computing. They try to provide a tool that both programs itself and learns on its own. Neural networks are structured to provide the capability to solve problems without the benefits of an expert and without the need of programming. They can seek patterns in data that no one knows are there.

A comparison of artificial intelligence's expert systems and neural networks is contained in Table 2.6.2.

Characteristics	**Von Neumann Architecture Used for Expert Systems**	**Artificial Neural Networks**
Processors	VLSI	Artificial Neural

	(traditional processors)	Networks;
		variety of technologies;
		hardware development is on going

Processing Approach	Separate	The same
Processing Approach	Processes problem rule at a one time; sequential	Multiple, simultaneously
Connections	Externally programmable	Dynamically self programming
Self learning	Only algorithmic parameters modified	Continuously adaptable
Fault tolerance	None without special processors	Significant in the very nature of the interconnected neurons
Neurobiology in design	None	Moderate
Programming	Through a rule based complicated	Self-programming; but network must be set up properly
Ability to be fast	Requires big processors	Requires multiple custom-built chips

Table 2.6.2 Comparisons of Expert Systems and Neural Networks.

Expert systems have enjoyed significant successes. However, artificial intelligence has encountered problems in areas such as vision, continuous speech recognition and synthesis, and machine learning. Artificial intelligence also is hostage to the speed of the processor that it runs on. Ultimately, it is restricted to the theoretical limit of a single processor. Artificial intelligence is also burdened by the fact that experts don't always speak in rules.

Yet, despite the advantages of neural networks over both expert systems and more traditional computing in these specific areas, neural nets are not complete solutions. They offer a capability that is not ironclad, such as a debugged accounting system. They learn, and as such, they do continue to make "mistakes." Furthermore, even when a network has been developed, there is no way to ensure that the network is the optimal network.

Neural systems do exact their own demands. They do require their implementor to meet a number of conditions. These conditions include:

- a data set which includes the information which can characterize the problem.
- an adequately sized data set to both train and test the network.
- an understanding of the basic nature of the problem to be solved so that basic first-cut decision on creating the network can be made. These decisions include the activization and transfer

functions, and the learning methods.

- an understanding of the development tools.
- adequate processing power (some applications demand real-time processing that exceeds what is available in the standard, sequential processing hardware. The development of hardware is the key to the future of neural networks).

Once these conditions are met, neural networks offer the opportunity of solving problems in an arena where traditional processors lack both the processing power and a step-by-step methodology. A number of very complicated problems cannot be solved in the traditional computing environments. For example, speech is something that all people can easily parse and understand. A person can understand a southern drawl, a Bronx accent, and the slurred words of a baby. Without the massively paralleled processing power of a neural network, this process is virtually impossible for a computer. Image recognition is another task that a human can easily do but which stymies even the biggest of computers. A person can recognize a plane as it turns, flies overhead, and disappears into a dot. A traditional computer might try to compare the changing images to a number of very different stored patterns.

This new way of computing requires skills beyond traditional computing. It is a natural evolution. Initially, computing was only hardware and engineers made it work. Then, there were software specialists - programmers, systems engineers, data base specialists, and designers. Now, there are also neural architects. This new professional needs to be skilled different than his predecessors of the past. For instance, he will need to know statistics in order to choose and evaluate training and testing situations. This skill of making neural networks work is one that will stress the logical thinking of current software engineers.

In summary, neural networks offer a unique way to solve some problems while making their own demands. The biggest demand is that the process is not simply logic. It involves an empirical skill, an intuitive feel as to how a network might be created.
