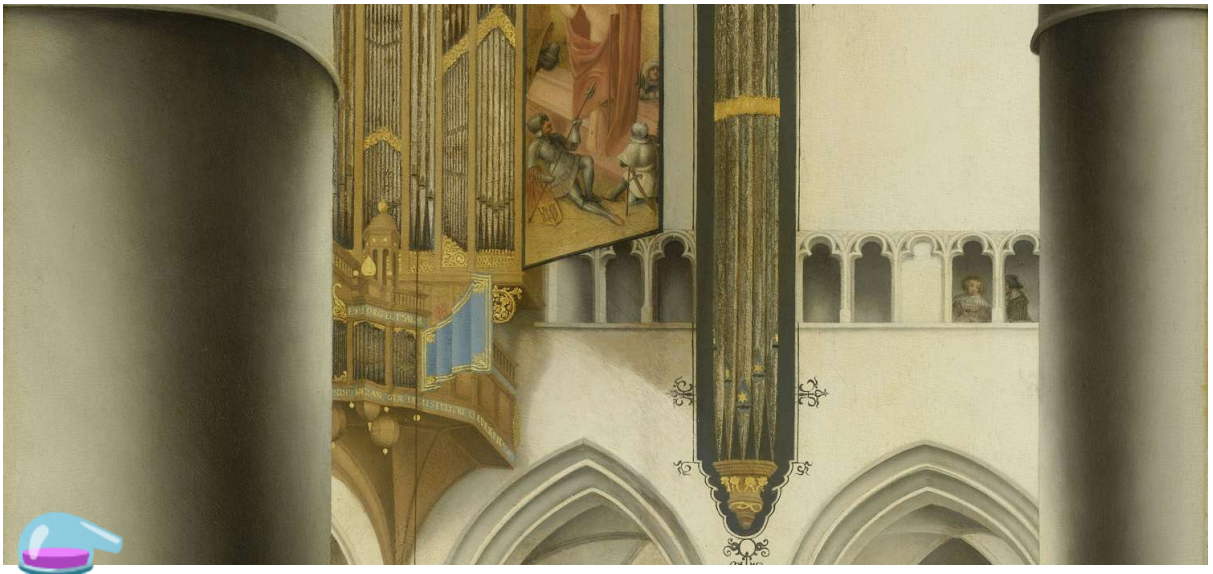
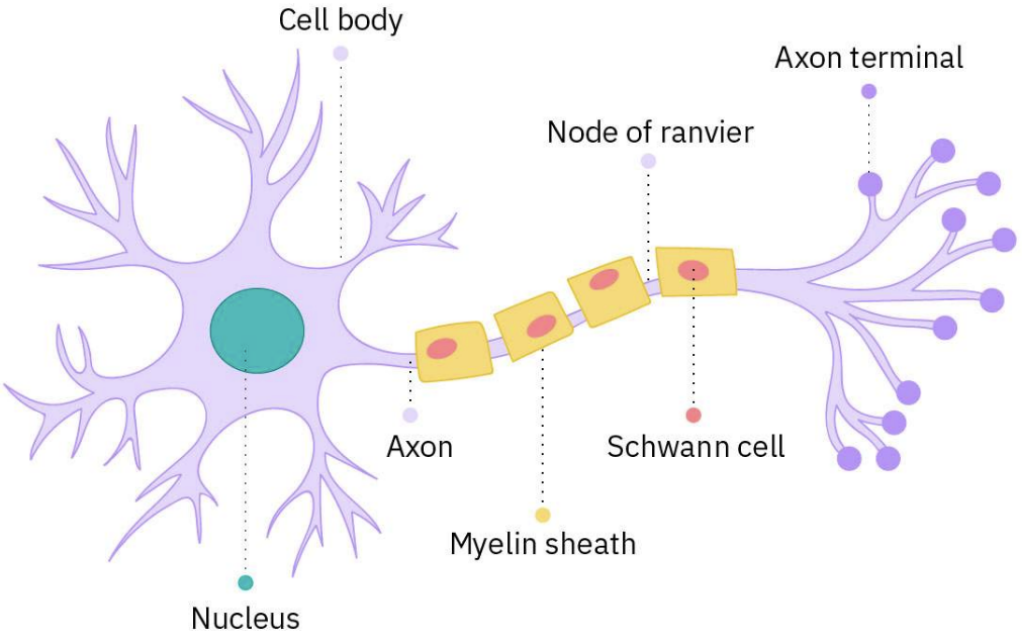


IBM ML/DL MOD3

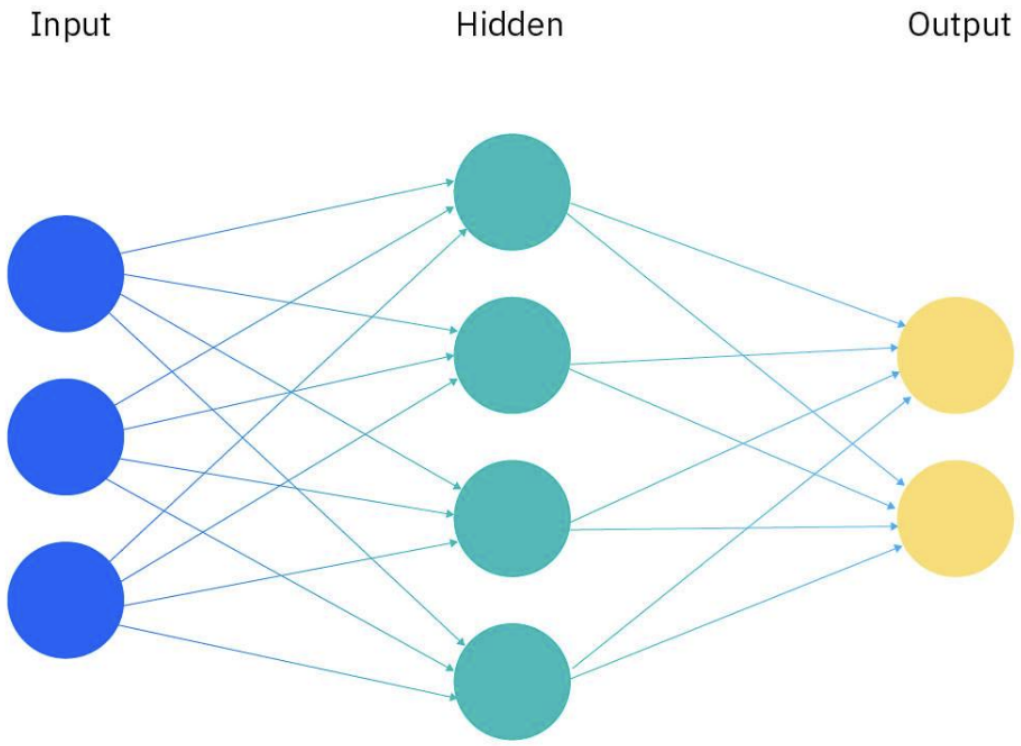
The deep learning ecosystem

Today, machine learning has evolved into a collection of powerful applications called the **deep learning ecosystem**. The foundation for many applications is called a **neural network**. A neural network uses electronic circuitry inspired by the way neurons communicate in the human brain.

HUMAN NEURON CELL

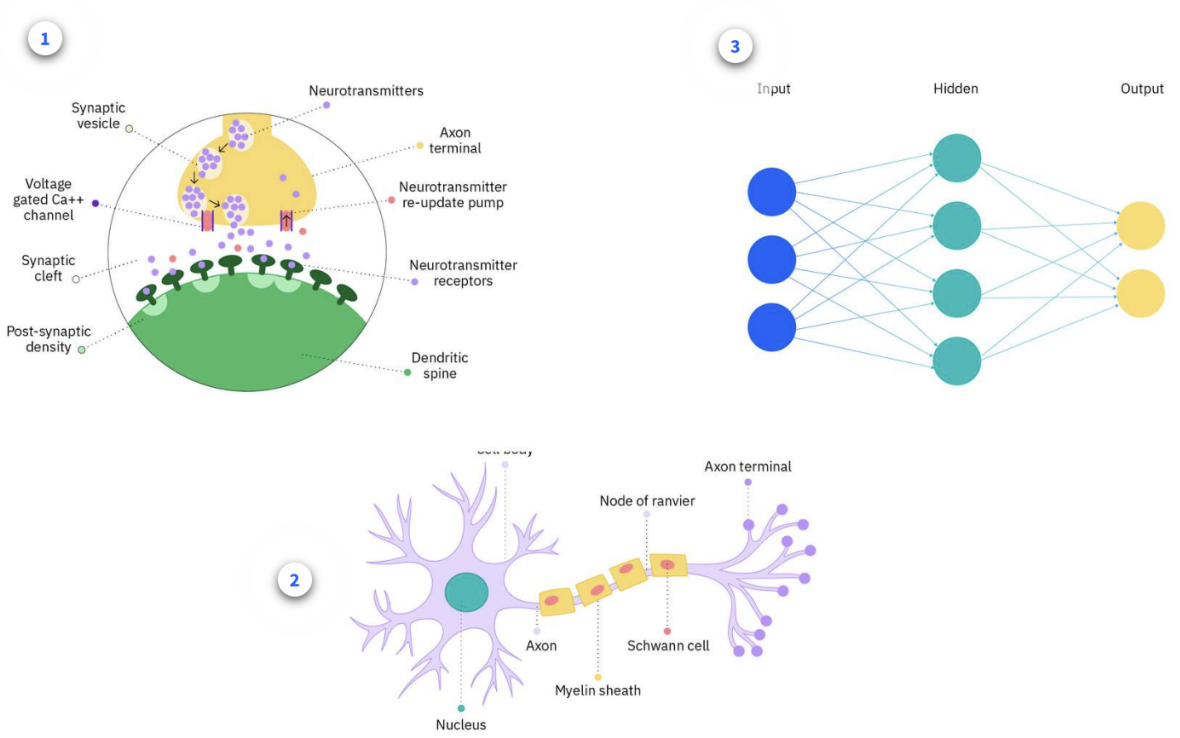


In the brain, cells called neurons have a cell body at one end where the nucleus resides, and a long axon leading to a set of branching terminals at the other end. Neurons communicate to each other by receiving signals into the axon, altering those signals, then transmitting them out through the terminals to other neurons. Researchers estimate that a human brain has about 100 billion neurons, each one connected to up to 10,000 other neurons.



Machine learning perceptron

In a neural network, a building block, called a **perceptron**, acts as the equivalent of a single neuron. A perceptron has an **input** layer, one or more **hidden layers**, and an **output** layer. A signal enters the input layer and the hidden layers run algorithms on the signal. Then, the result is passed to the output layer.



The hidden layers in a neural network resemble, as a group, the long cell body that connects dendrites to axons within a human brain cell. Those hidden layers contain **nodes**. Each node runs an algorithm and bits of additional code to test and adjust its result. When the value reaches a certain threshold, the node “fires”.

Note: A node often uses a sigmoid function to determine whether or not to “fire”. As explained previously, a sigmoid function can generate a binary answer, such as, YES or NO. The binary answer tells the node whether or not to fire. You can think of the threshold as a hurdle a solution must jump over to give a result of YES.

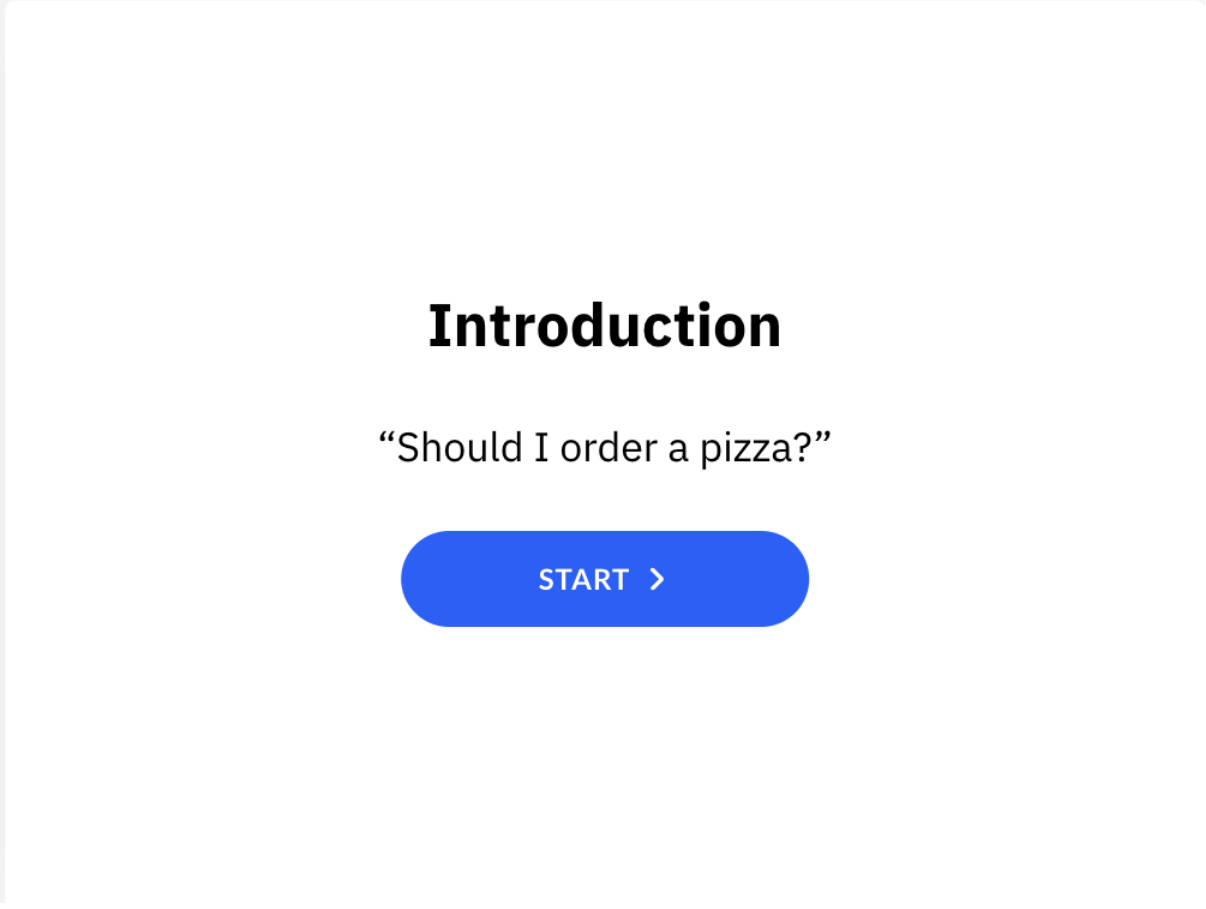
If there are other nodes connected to the node, they are activated when the signal reaches them. If the other nodes reach their own thresholds, then they fire. The signal cascades down through the hidden layers in a way that’s somewhat similar to how a signal passes down the body of a human brain cell.

Keep in mind that these resemblances are only similarities. Neural networks are inspired by the human brain, but the activities inside neural networks are quite different.

A path through a neural network

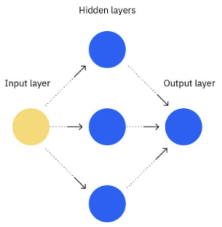
The operation of a neural network is pure mathematics. The network isn't "thinking"; it is calculating. But it's using those calculations to create an output that humans can interpret as an answer or a recommendation.

> For example, consider the following scenario: Marc is trying to decide if he should order a pizza. Marc wonders how a neural network might respond to the question, "Should I order a pizza?" The following scenario provides the step-by-step flow of decision making as individual nodes perform algorithms and pass on their results.



Step 1

Should I order a pizza? I'll ask the neural network.



This neural network has already been trained. It has learned many things about Marc's past pizza buying behavior, so it's ready to suggest whether Marc is likely to order one today. Today, the neural network is going to invoke three things it has observed:

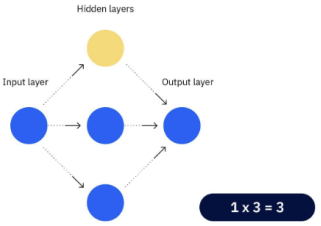
1. Marc usually orders ahead.
2. Marc is trying to lose weight, so lately he's ordered pizza less often.
3. Marc's email includes a coupon for a free pizza.

Now Marc asks, "Am I likely to order a pizza today?"

1 2 3 4 5 ✓

Step 2

Ordering ahead?



One node in the hidden layer looks up Marc's buying pattern and observes that Marc always orders ahead. So, the node assigns ordering ahead a value of 1. (For clarity, one (1) and zero (0) values are used.)

The system has learned something else about Marc's ordering: Marc gives more **weight** to some factors than others. So, the system gives each factor a weight based on Marc's past behavior. The more Marc tends to do something, the higher the weight.

Marc always orders ahead, so the system gives today's order-ahead factor a weight of 3. Multiplying the initial value (1) by the weight (3) gives the node an output value of 3.

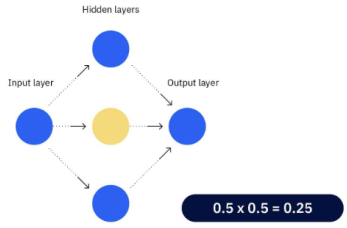
So far, the total value of the ordering factor is:

1 x 3 = 3

1 2 3 4 5 ✓

Step 3

Getting salad instead?



The next node in the hidden layer registers that, lately, Marc has been ordering a salad about half the time. So, the node gives the salad option an initial value 0.5.

The system also knows that Marc is not consistent about avoiding pizza for his diet, so it gives the option to order a salad a weight of 0.5.

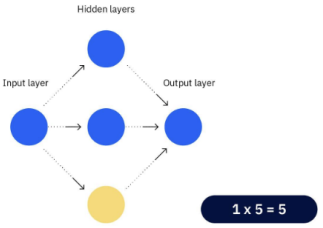
The initial value (0.5) multiplied by the weight (0.5) is 0.25.

0.5 x 0.5 = 0.25

1 2 3 4 5 ✓

Step 4

Free coupon?



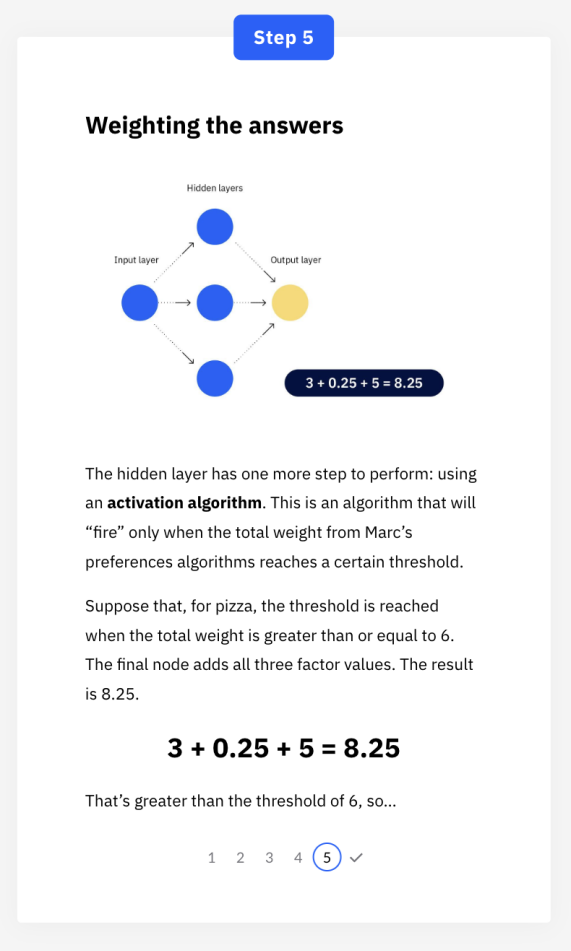
The third node in the hidden layer knows that Marc has a coupon for a free pizza. So, it gives the coupon a factor of 1.

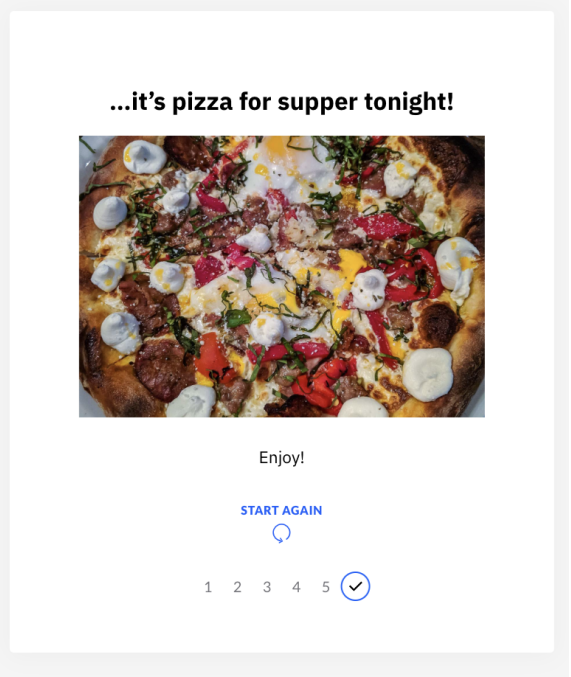
Finally, the system observes that Marc always uses free coupons as fast as they arrive. So, the system multiplies the coupon factor by a weight of 5:

1 x 5 = 5

With these algorithms now run, the system advances results from all three hidden layer nodes to the output node.

1 2 3 4 5 ✓

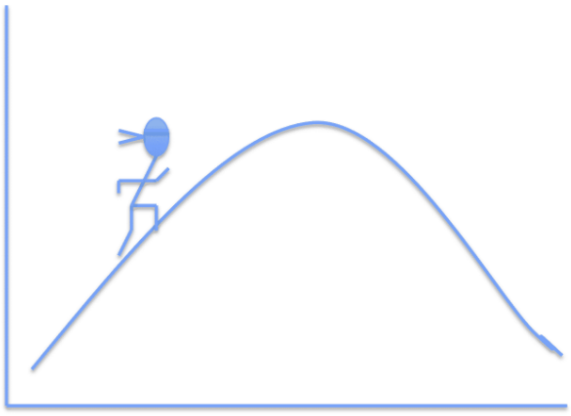




Machine learning is often trial and error

Previous lessons in this course covered how a neural network makes decisions based on what it learns. But you might still be wondering, how does a neural network learns in the first place. The answer is: by continuously adjusting itself, in a process that humans refer to as **trial and error**.

Once a neural network has ingested or already learned a certain amount of data, it stores the data in its "body of information", called its **corpus**. In order to learn, the neural network constantly tests new data or the results of its calculation against its corpus. If the network determines that the new data or results don't match the patterns it has already established, it modifies those patterns for a better fit. Sometimes, to improve a single match, the network tests hundreds or thousands of modifications very rapidly and makes adjustments. Then, the network tests to determine if the match is improving. So, step by step, the machine learns.



To visualize this, imagine that someone blindfolds you. They tell you that your assignment is to climb a hill and reach its exact top using the fewest number steps possible. Then, they send you walking up the hill.

Machine learning makes many guesses

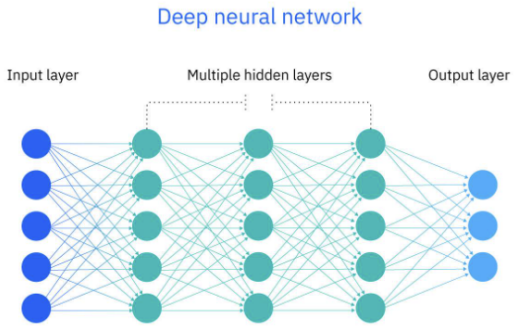
Machine learning uses its tremendous calculation speed to make many guesses that bring it closer and closer to an answer. It randomly makes its first guess, sets that guess as a variable, then tests how accurately the guess fits with both old and new data. Next, it makes an adjustment to the variable and tries again. Using mathematical processes to help it choose right-size adjustments, the system keeps on trying, getting closer and closer to perfection but never quite reaching it.

For this reason, many AI systems output a confidence value along with an answer or prediction. For example, a system predicting effective treatments for a cancer patient might output two or three suggested approaches, along with a measure of how confident it is that each treatment might work. This reflects how the system reaches those decisions. The system also leaves the final decision to the doctor who knows the patient.

Any computer can perform at least a crude kind of machine learning based on cycles of estimation. Classical machine learning can do it, too. But depending on the complexity of a problem, a conventional computer or even a classical system might take days (or centuries!) to reach a conclusion.

In many modern applications of AI, the unstructured data involved is complex enough to overwhelm even a simple perceptron, such as to decide whether to order pizza in a previous lesson. So, a perceptron requires more **brainpower** in the form of **deep learning**. Deep learning relies on multiple layers of nodes (even multiple groups of perceptrons with multiple layers of nodes!) to finish the work in reasonable time.

From perceptrons to deep learning



In a previous lesson, Marc ordered a pizza using a perceptron in which one hidden layer of nodes did most of the work. But advanced AI systems use many hidden layers whose algorithms pass the results of sophisticated calculations. This is a **deep neural network (DNN)**. DNN layers can be arranged in groups or elaborate blocks of groups for greater power. DNNs can even be doubled in competing teams that judge and learn from each other's mistakes, without human intervention. This creates powerful reinforcement learning.