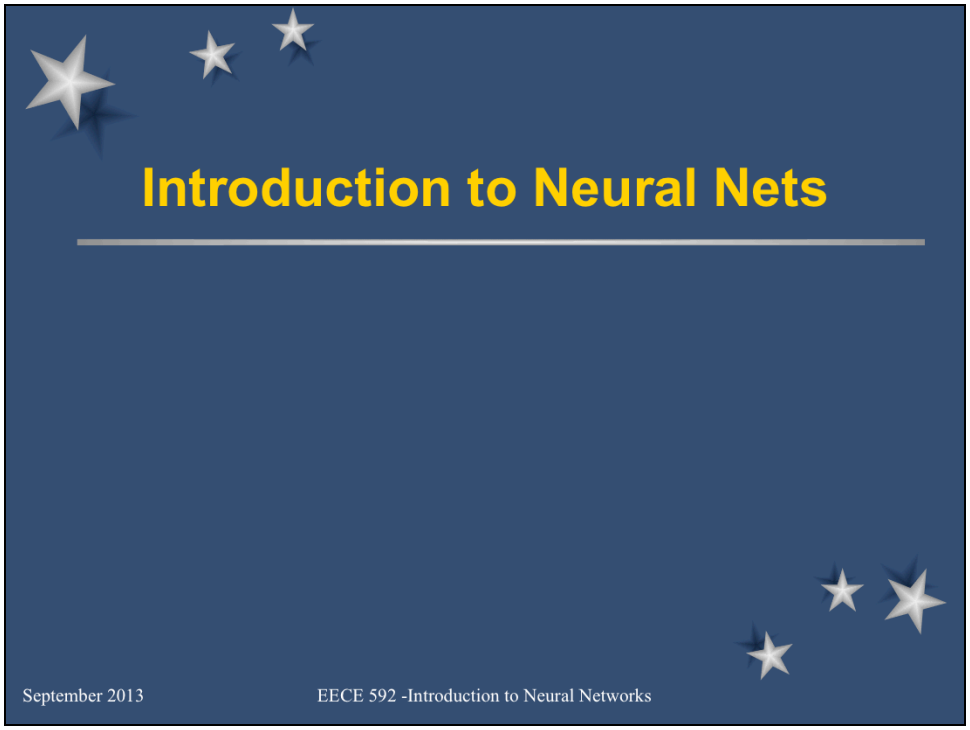
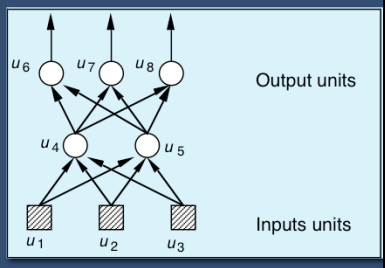

Introduction to Neural Nets
September 2013
EECE 592 -Introduction to Neural Networks

Introduction to Neural Nets

- A neural network model is a set of *cells*
- The connection between units is *weighted*
- *weights* determine the characteristics of the model



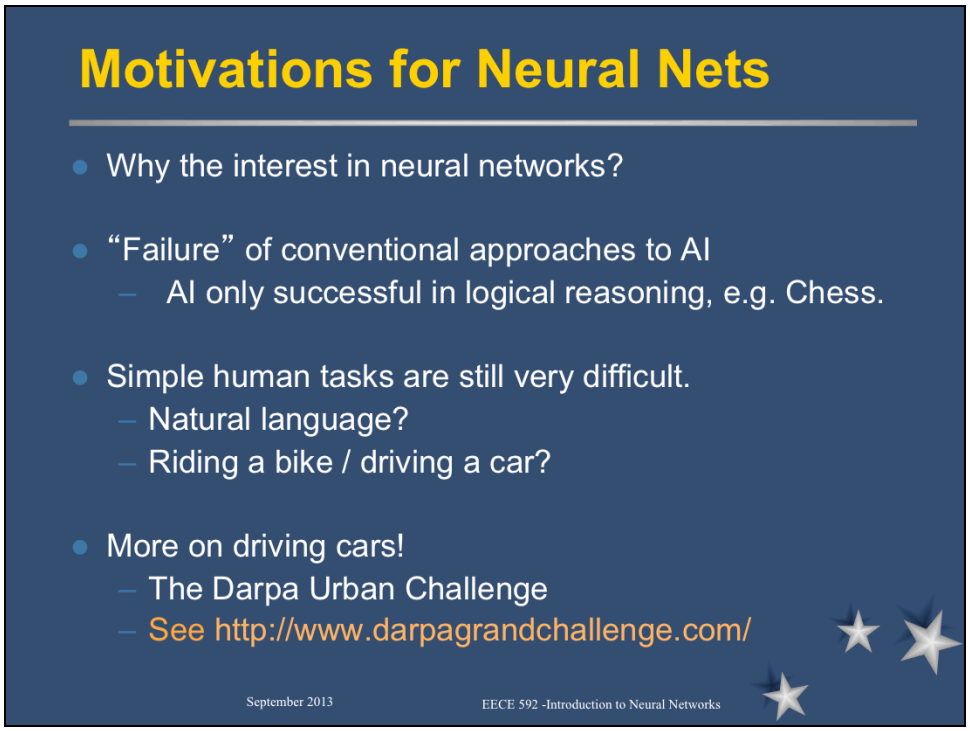
A neural network model is a set of *cells* or units which each generate an output value or *activation* determined by the sum of the inputs to the unit.

Each activation presents itself as an input to the other units.

The connection between units is *weighted* so that the significance of each input to a unit is either emphasized or inhibited.

It is these *weights* which determine the characteristics of the model and it is by modification of these weights that different network models can be made to change their behaviour to suit different applications.

Note that units U_1, U_2 and U_3, are not neurons at all but instead represent input values.



Study in the field of connectionist models increased rapidly over the mid 80s to late 90s - why is this so?

Conventional approaches to AI

The aim of AI has been to make computers perform as well as humans.

However, no general purpose approaches exist today which show any signs of being able to perform tasks such as speech recognition, natural language understanding or vision.

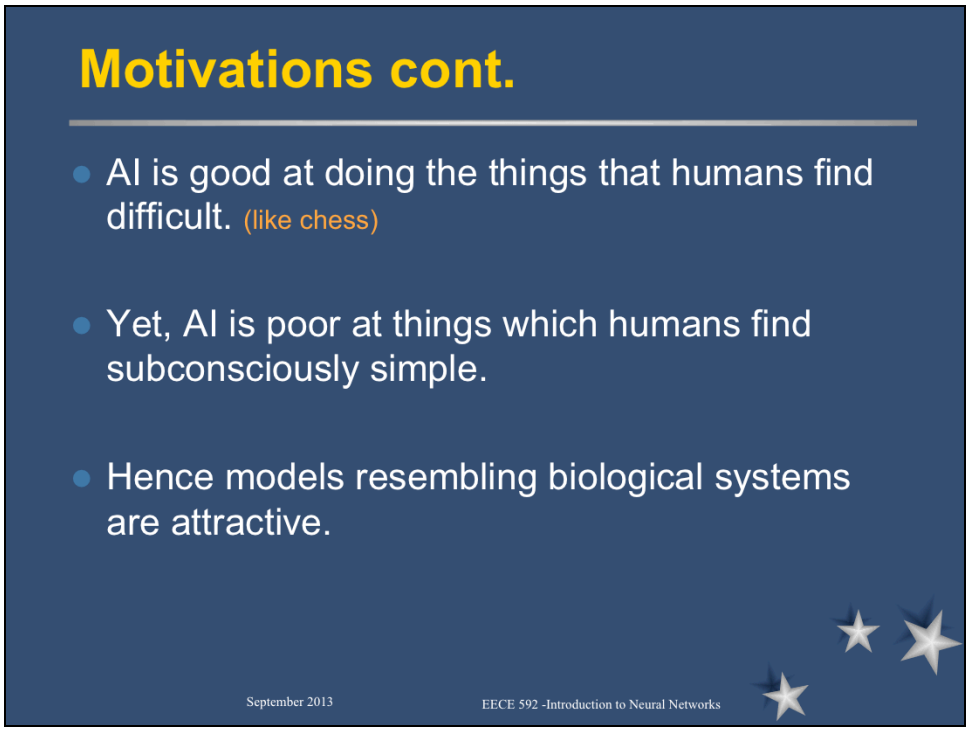
This goal has not yet been met and it does not look likely that conventional AI holds any promising future.

Simple human tasks

Whereas AI has been able to provide successful logical reasoning in limited domains such as used by chess machines or expert systems, similar approaches have not been successful in tackling seemingly simple tasks that humans perform everyday. They are said to be "brittle". Walking, listening, seeing we all can do with almost no conscious effort at all.

How about driving a car through the rain? You have to process a continuously changing image, distorted by drops of rain while at the same time adjust the steering to keep the car on the road. Not to mention having to totally ignore the sweep of the wiper blades!

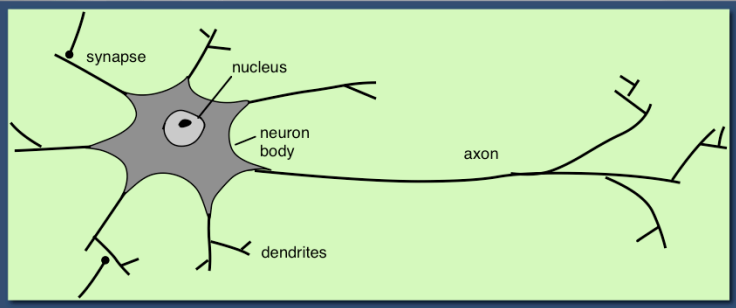
Autonomous driving was the topic of a few DARPA funded challenges in recent years. See http://www.darpa.mil/grandchallenge/index.asp



It seems that AI has been good at doing the things that humans find difficult, but very poor at those which humans find subconsciously simple.

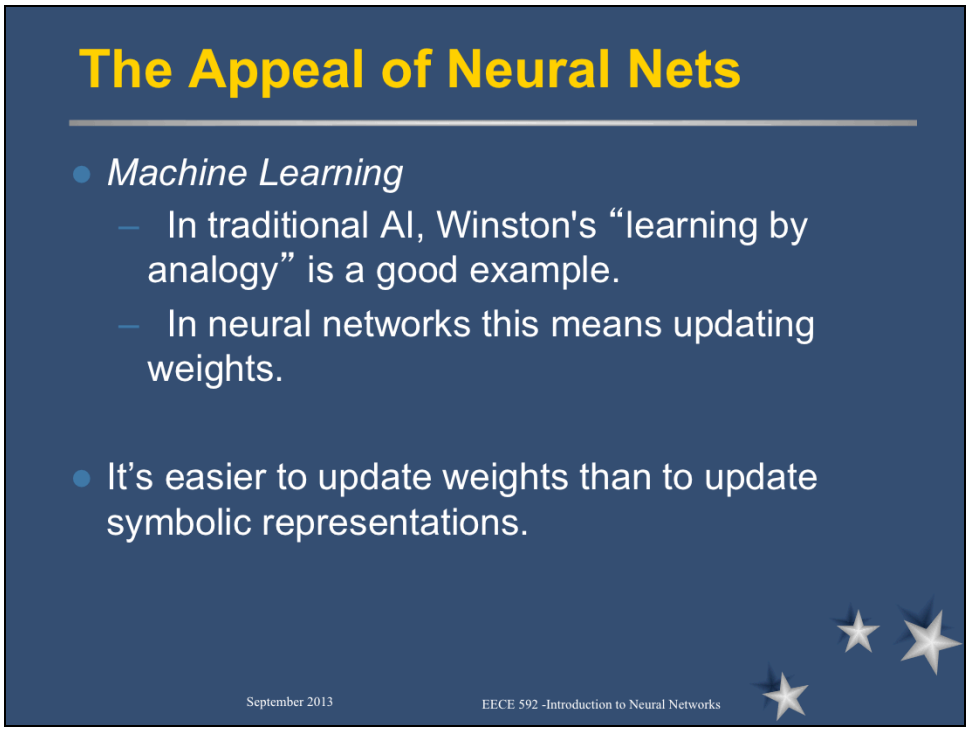
Thus it is natural to look at models which in some way do resemble biological systems, in the hope that by devising models which qualitatively possess similar characteristics, similar behaviour can be emulated.

Biological Neurons



- The human brain is composed of 10^{11} of these cells.
  - that's *100 000 000 000*

This course does not discuss any further, how biological neurons work. This is a huge topic in its own right! However suffice to say that there is a lot published on the topic. For our purposes, it is useful to know that learning is believed to take place at the synapses. These are the points at which one neuron connects to another.



There are two aspects to connectionist models which are computationally quite appealing - learning and representation.

In neural networks this almost always means updating weights of inter unit connections according to some algorithm.

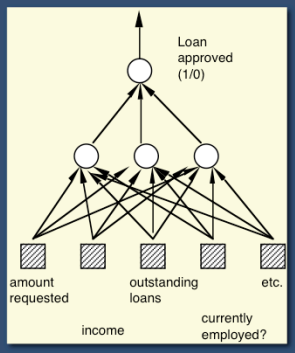
In traditional AI, machine learning involves updating representations based upon some input information. Winston's learning by analogy is a good example. However, the mechanisms to update symbolic representations are often complex in nature.

Neural network models are particularly suited to machine learning and compared to traditional AI enable learning via mathematically based algorithms, I.e. updating a bunch of numerical parameters using some formula.

The alternative are approaches where manipulation of potentially complex symbolic representations is necessary.

Appeal of NN cont.

- Knowledge Representation
  - Example - credit loan
  - The final output unit is 1 if loan approved, 0 otherwise.

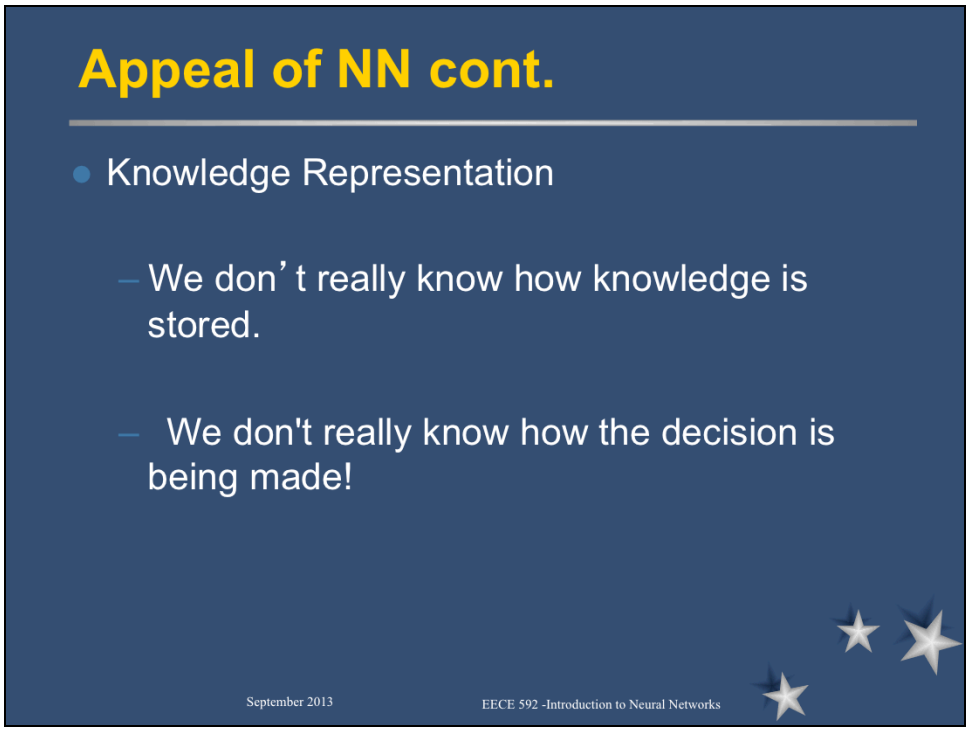


Knowledge Representation

Traditional AI has almost exclusively employed symbolic techniques as a means of knowledge representation. Knowledge representation in connectionist models is significantly different from such approaches. For example here is a credit loan example: the output is 1 if the loan is approved, 0 if not.

Note: One advantage of neural networks over polynomial curve fitting, in the case of multivariate non-linear functions is that the number of free parameters can be scaled linearly or quadratically according to the complexity of the problem, rather than exponentially with the dimensionality - d^M where d is the dimensionality of the input and M is the degree of the polynomial.

Ref: "Neural Networks for Pattern Recognition", Christopher Bishop. Pg 9, 15-16

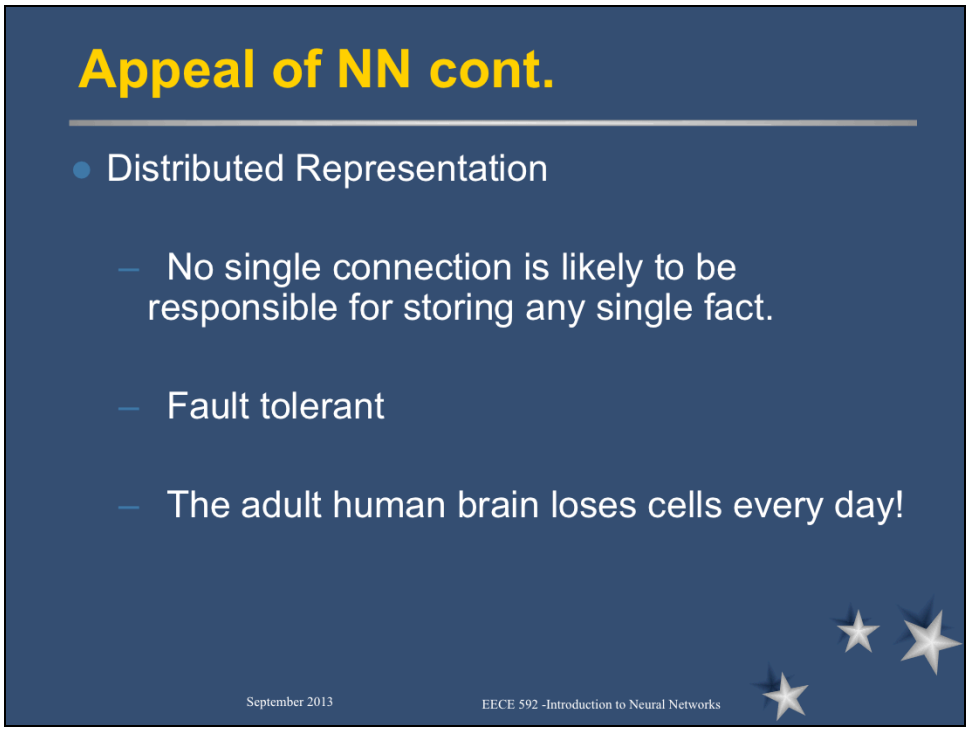


Knowledge Representation

The knowledge required to perform the decision making process is somehow stored in the weights belonging to the intermediate and output units of the network

However we don't really know what each of the units is storing and consequently we don’t know how the decision is being made!

This can be problematic. Say the loan applicant wishes to know why their application was rejected. The neural net cannot tell us why.



Distributed Representations and Data Redundancy

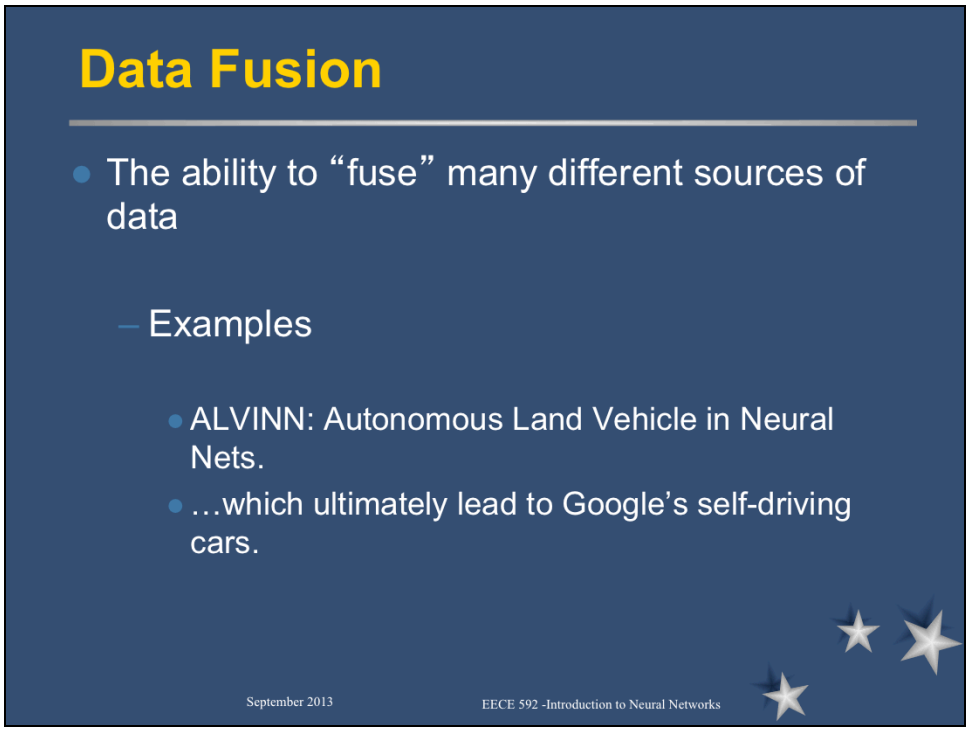
The set of weights form a distributed representation in which no single weight or unit is likely to be responsible for storing any single fact.

A property of such a representation is that it is very redundant and should a weight or a whole unit fail, the effect on the performance of the whole network should not be catastrophic.

In fact the adult human brain loses about 100,000 cells each day without any noticeable effect on the individual!

> Proof of this is the so-called "Grandmother cell" theory. This theory says that there is no such thing as a single brain cell that "fires" when you see and recognize your grandmother, say. If there was, then the death of this cell would mean you would no longer recognize your grandmother, literally overnight! Therefore, there must be many, many cells that support the recognition process.

Oh and in case you're worried, I figure that it will take you about 274 years before you've lost enough brain cells to be down to 90% of your original neuron count!

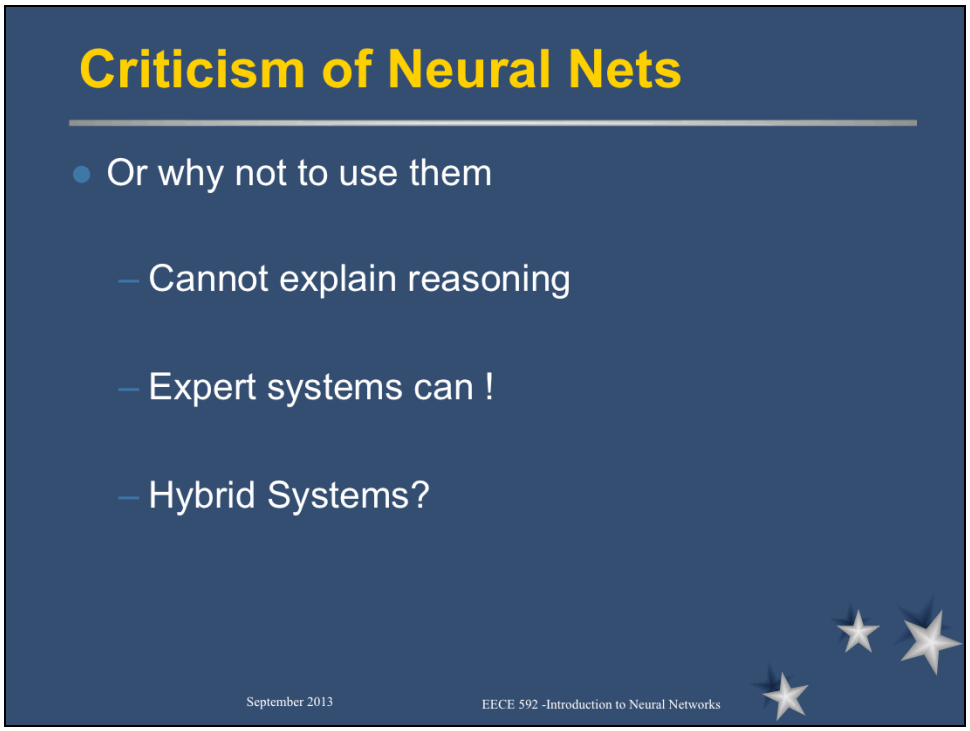


Connectionist models are highly suited to the integration of data from various sources such as auditory or visual.

Although the ALVINN project is now no longer active, it was an excellent early example of data fusion using neural nets. Its input included both visual information, from a camera mounted at the top of the vehicle and an infrared range finder. Both kinds of data were used to determine how much and in which direction to turn the steering wheel. Many of the DARPA Urban Challenge contestants also relied on multiple forms of input.

SebastianThrun's team, whose work included entries in the DARPA Challenges (2nd place winner in the DARPA Urban Challenge) lead to the development of the Google self-driving cars!

http://spectrum.ieee.org/automaton/robotics/artificial-intelligence/how-google-self-driving-car-works



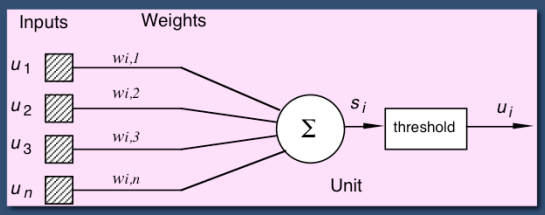
A criticism of connectionist models is their inability to explain the reasoning for a result in a useful way. This is perhaps the single most significant criticism of connectionist approaches. There are many situations in which not being able to explain a decision would be prohibitive. E.g. the loan application example. A customer would want to know the reason for refusal - too much existing debt, inadequate salary, bad credit history etc.

Expert systems on the other hand, by the explicit use of rules are easily capable of providing explanations for their inferences. We will touch upon rule based systems later on in the course.

To perhaps address this concern, attempts have been made to combine the natural ability of neural nets to handle large amounts of data with the inferencing ability of expert systems.

A Neuron

- A "neuron"
  - The McCulloch and Pitts (1943) model



  - weighted sum

$$S_i = \sum_{j=0}^{n} w_{ij} u_j$$

The typical model used today still conforms to that suggested by McCulloch and Pitts (1943).

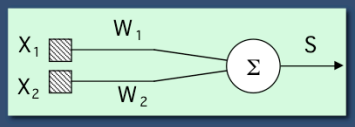
The ith unit u performs a weighted sum of its inputs (which may also be the outputs of other units).

$$S_i = \sum_{j=0}^{n} w_{ij} u_j$$

Not shown is a bias, which is assumed to be a weight from an input that is always 1.

What does a neuron do?

- Consider a 2-dimensional input

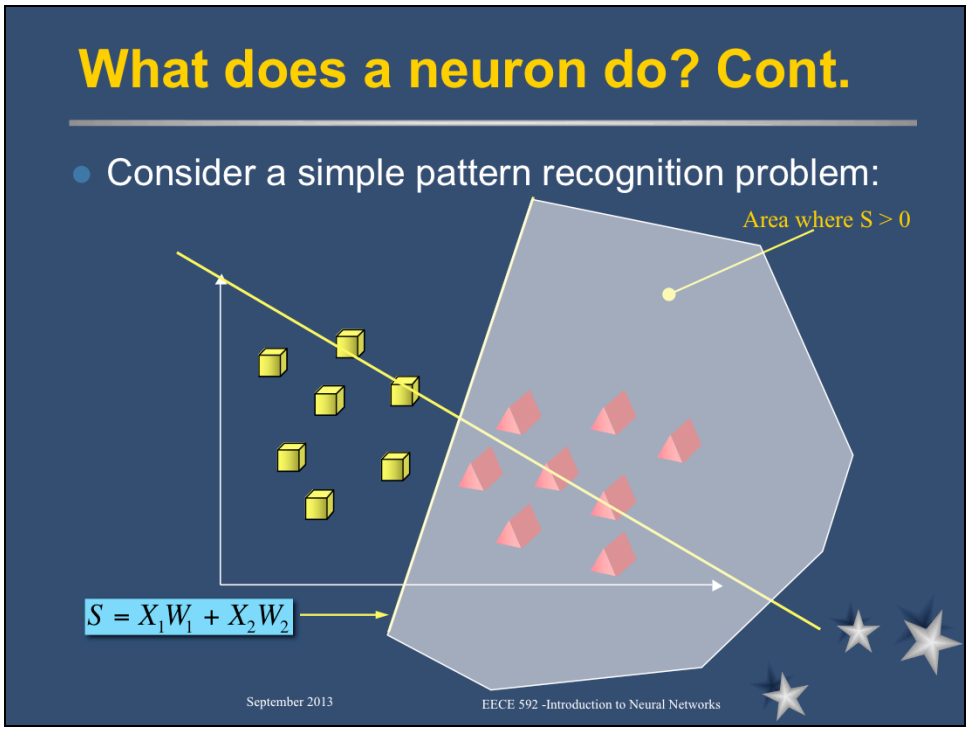


  - The output is: $S = X_1W_1 + X_2W_2$
    - this is the equation of a straight line - important

For a two dimensional problem (i.e. one in which there are just two components in the input vector), we note that the neurons defines a straight line. This is a decision boundary.

Can you guess what the neuron is defining in the case of a three dimensional input?

In general then, for *n* dimensional inputs, the neuron will define a decision boundary that is a hyper plane in *n-1* dimensions.



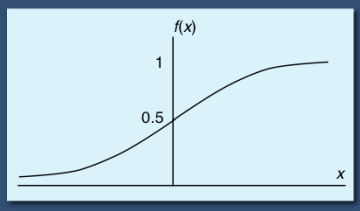
Adjusting the weights of a neural network causes a corresponding change in the orientation of a *decision boundary*. A step threshold function will return 0 for any point on one side of the line and 1 for any point on the other (as illustrated by the shaded area). For the neuron to successfully categorize all squares as 0 and all triangles as 1 say, the learning algorithm must find a line (set of values for W_1 and W_2) which correctly separate all inputs presented during learning.

Activation Functions

- Activation functions $f()$
  - This may be a simple threshold e.g.

$$f(x) = \begin{cases} +1 & for \quad x>0 \\ 0 & for \quad x \le 0 \end{cases}$$

  - or some more complex *squashing function* e.g.



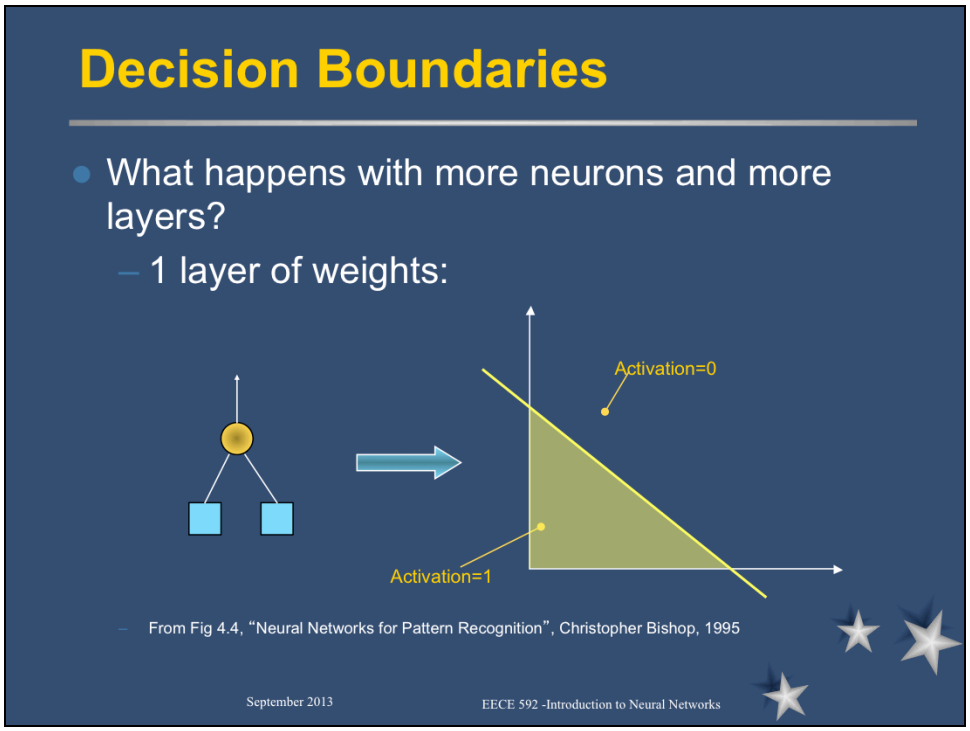
Activation functions

The weighted sum S_i of the unit is subjected to a threshold through some *activation function* $f()$. This may be a simple step threshold function e.g.

$$f(x) = \begin{cases} +1 \text{ for } x > 0 \\ 0 \text{ for } x \le 0 \end{cases}$$

or some more complex *squashing function* such as the sigmoidal:

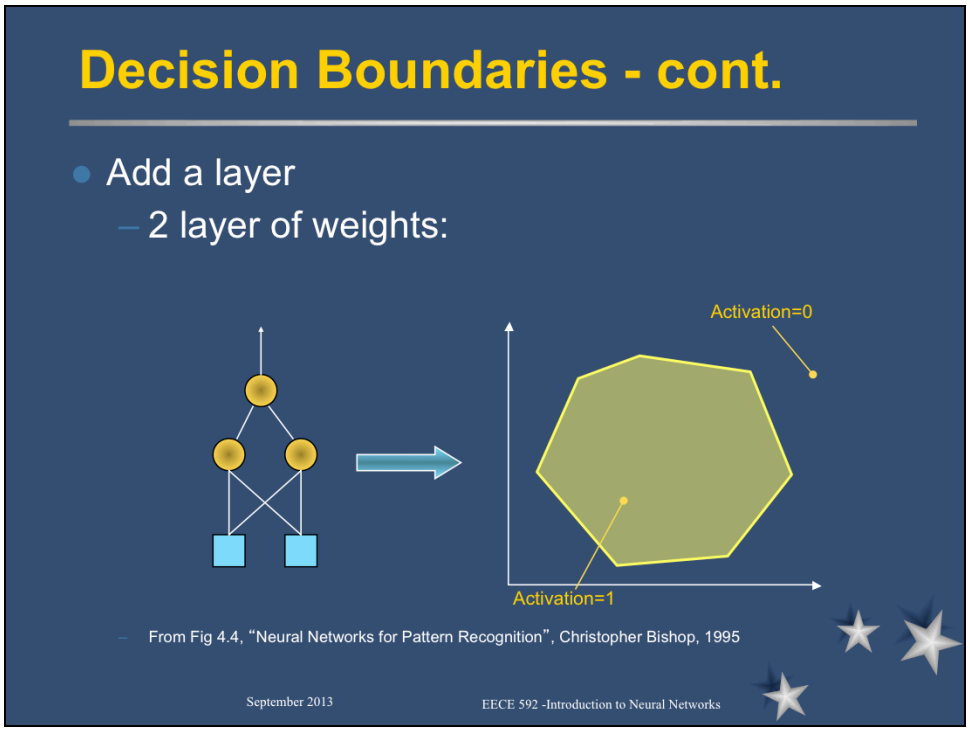
$$f(x) = \frac{1}{1+e^{-x}}$$



With a single layer of weights, a feed-forward neural net will have a single decision output neuron and therefore a single decision boundary.

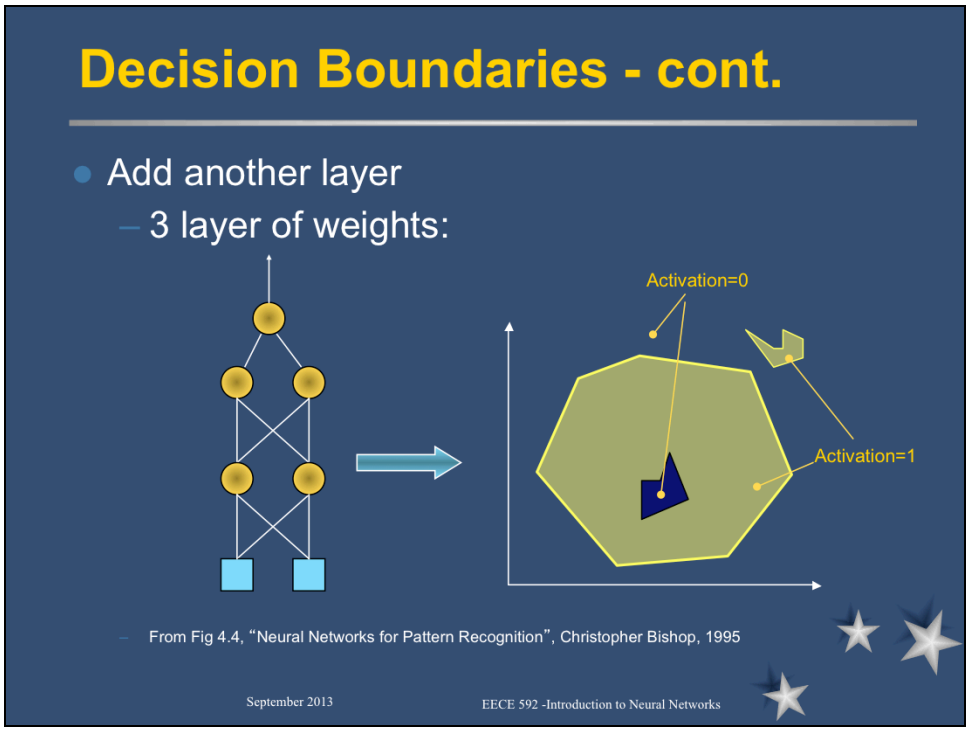
The dimensionality of the decision boundary is determined by the number of input variables in the input vector. In the example illustrated, a two-dimensional input gives rise to a linear decision boundary.

Notice then that a single layer of weights is only able to solve linear problems. I.e. categorisation problems that can be separated using linear decision boundaries.



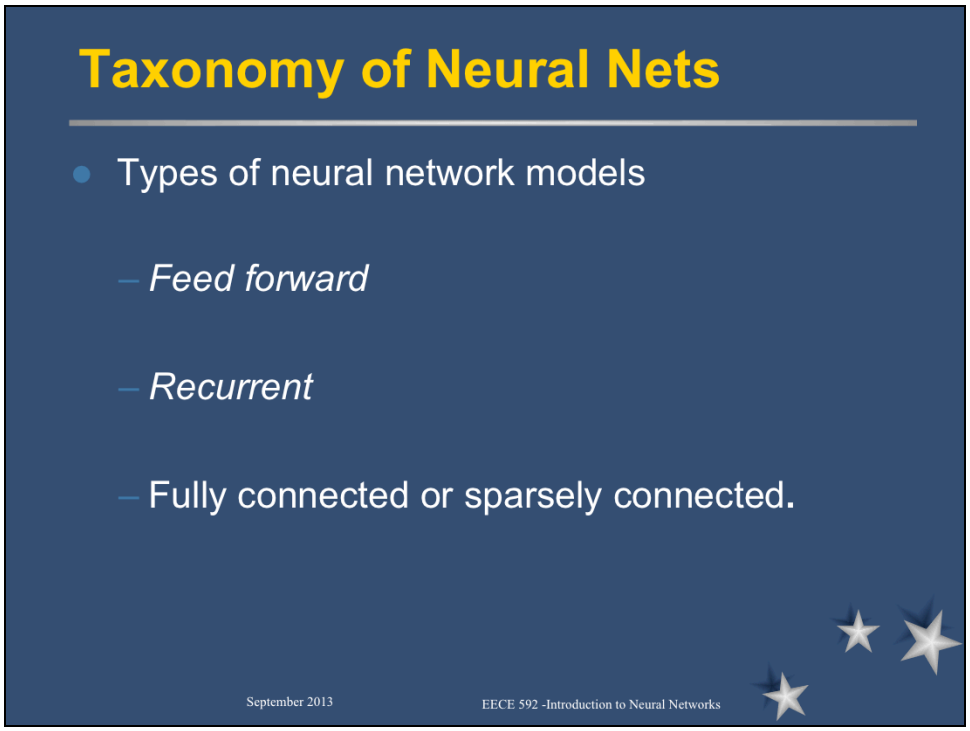
A second layer of weights means that many decision boundaries can be combined to form closed areas. For a two dimensional input this translates to areas. Can you guess what the decision boundaries for a three dimensional input give rise to? For input vectors with greater numbers of inputs, decision boundaries form hyper volumes.

Two layers of weights are capable of treating non-linearly separable problems.



Three layers of weights enable multiple disjoint regions to form. Theoretically, three layers of weights should be able to solve any learning problem. In practise, training a three layer network is computationally expensive and may not often be necessary.

Update: Sept 2013 – Actually, according to the "Universal Approximation Theory" it is possible to model any arbitrary continuous function that maps a set of input real numbers to some output set of real numbers using a single layer of hidden neurons. (http://en.wikipedia.org/wiki/Universal_approximation_theorem).

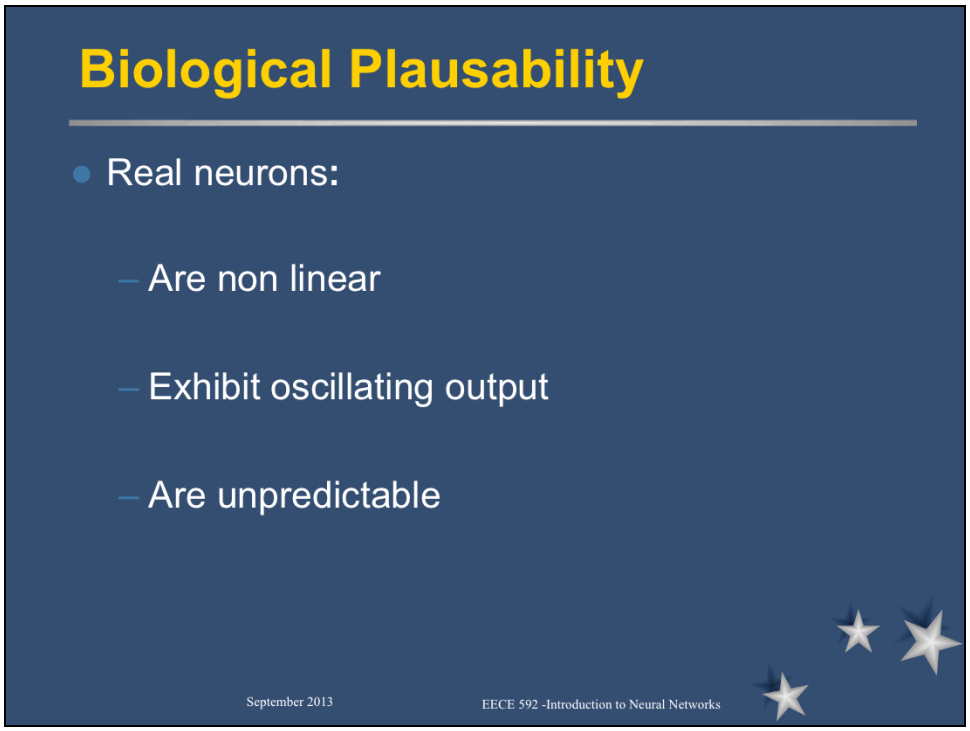


Two basic paradigms for network models can be observed.

Models may be *feed forward* networks in which the output of one set of units is fed into another *layer* of units. (A multi-layer perceptron)

Networks can be *recurrent* networks in which the output of a unit as well as being an input to other units is also an input to itself. (E.g. the Boltzmann Machine)

Networks may either be fully connected or sparsely connected. (Used in so-called convolution networks).



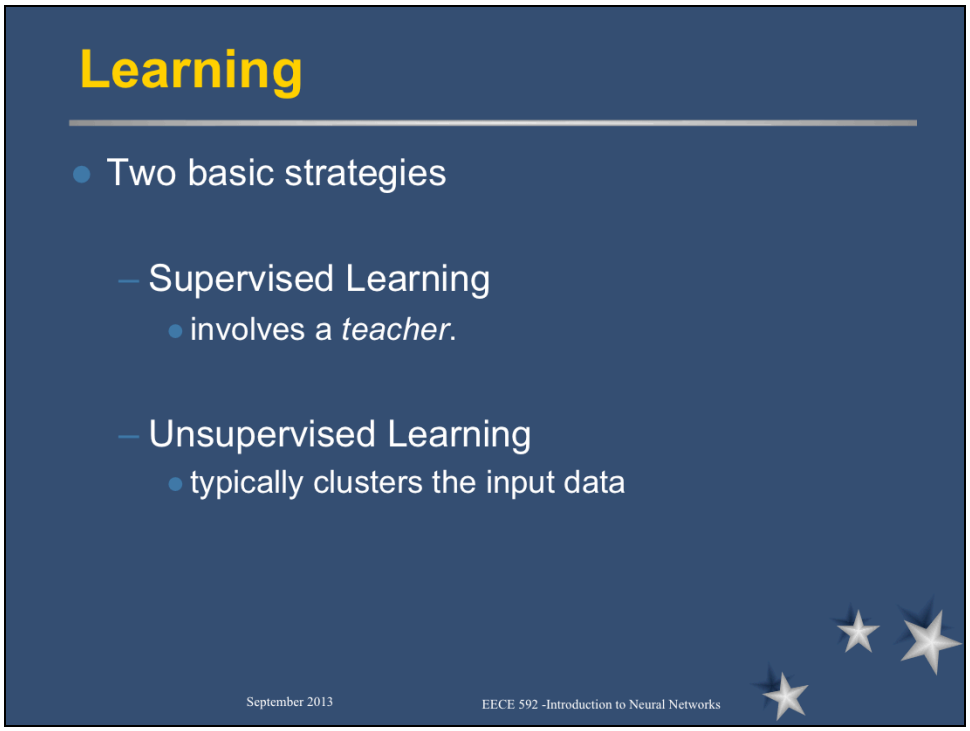
Although neural networks do appeal for their biological realism, they do not rate highly in their direct application to brain modelling. When compared one-on-one, artificial neurons do not compare favourably with their natural counterparts.

Real neurons:

- perform a non linear summation of their inputs.
- produce a sequence of pulses not a simple output level.
- the amount of neurotransmitter released at a synapse may vary unpredictably.

Artificial neurons:

- Are linear
- Output is typically steady
- Are deterministic (with the exception of the Boltzmann machine)



Broadly speaking there are two categories of learning, *supervised* and *unsupervised*.

Supervised Learning

For supervised learning, each example E^k is associated with some correct response C^k. In supervised learning, the required response C^k acts as *teacher*. K is the number of patterns or examples that make up the training data.

The learning algorithm must attempt to adjust all weights and biases in the network so that presentation of each input pattern E^k, causes the network to generate on the output, the desired or trained output, C^k.

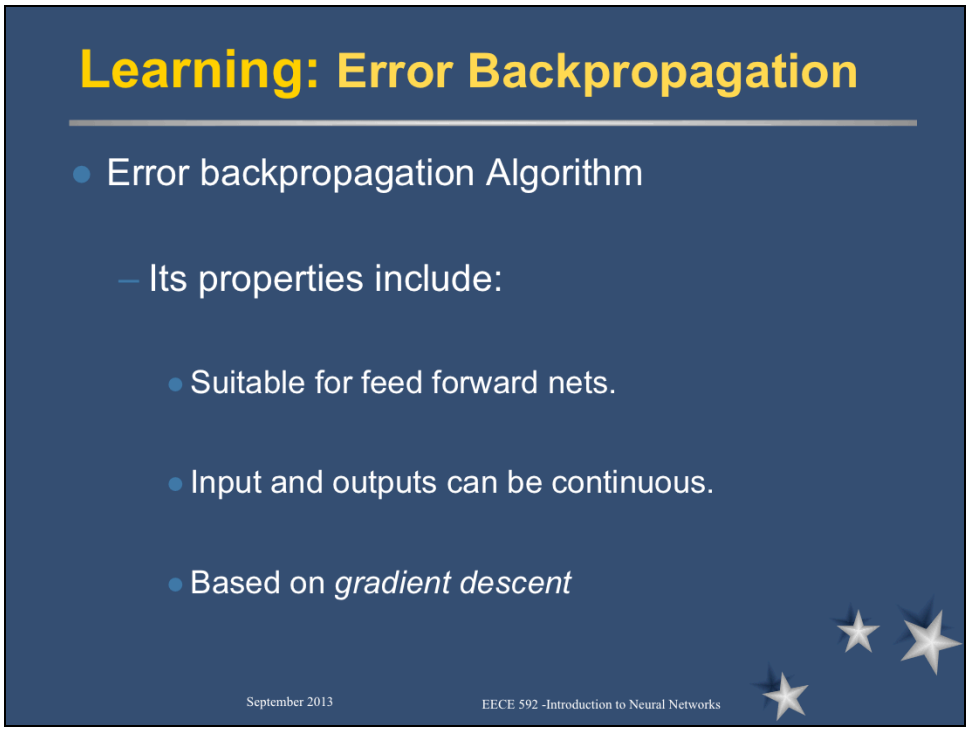
Unsupervised Learning

Unsupervised learning algorithms do not require a teacher and hence there are no correct response vectors C^k.

In general, unsupervised learning algorithms attempt to cluster or categorize the input data.

One of the most well known unsupervised learning paradigms is the self-organising feature map. This will be covered later in this course.

There is also another approach to unsupervised learning, in which the purpose is to learn internal representations using auto-encoders. These are used in "Deep Neural Networks"



The Error Backpropagation algorithm or *backpropagation* (or just BP) as it is often shortened to, is one of the most popular learning algorithms used in neural net learning. Its properties include:

- Used with feed forward multi-layer perceptrons.
- Input and outputs can be continuous.
- Based on a technique known as *gradient descent* - fundamental to many connectionist learning approaches.

Gradient descent is a mathematical term describing general class of algorithms which implement a step by step decay in one variable given a change in another controlling variable.

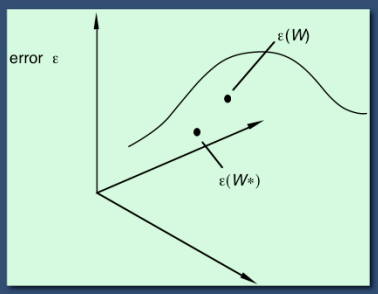
Gradient Descent

- Thus in changing the weights from

$$W \rightarrow W^*$$

- the error changes

$$\varepsilon(W) \rightarrow \varepsilon(W^*)$$



Given a weight vector W and a measure of the error for those weights $\varepsilon(W)$, we would like an algorithm that calculates W^*, the new weight vector whose error is less than that of W.

Thus in changing the weights from

$$W \rightarrow W^*$$

the error changes

$$\varepsilon(W) \rightarrow \varepsilon(W^*)$$

I.e.

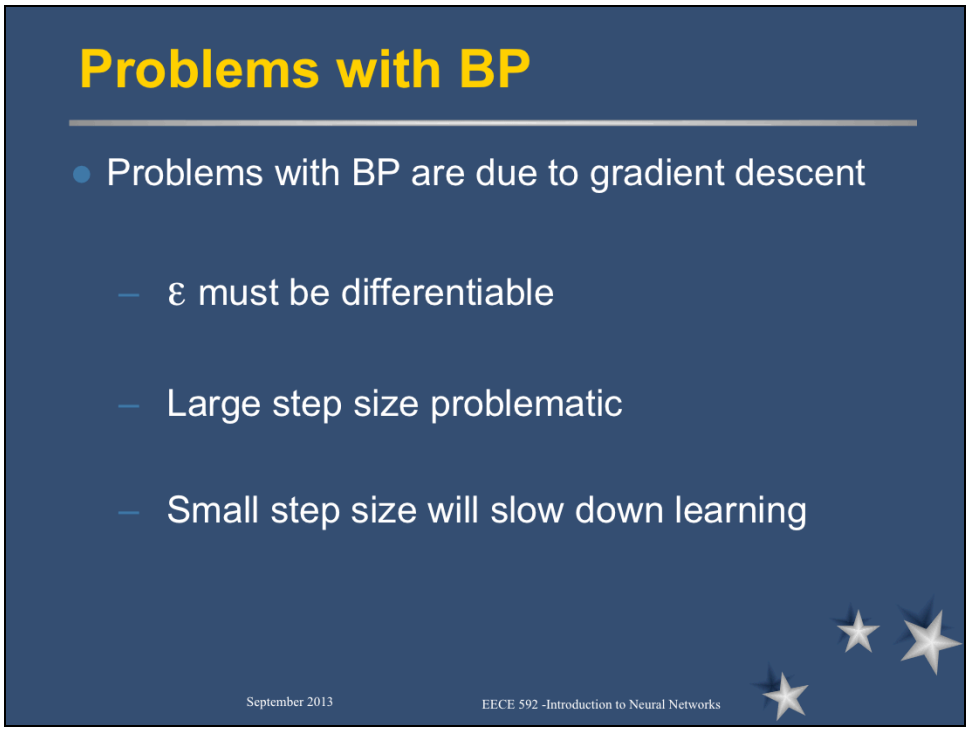
$$W^* - W \propto -\nabla\varepsilon(W)$$

$$W^* = W - \rho\nabla\varepsilon(W)$$

or

$$\nabla W \propto -\nabla\varepsilon(W)$$

In a later class, we will derive the learning algorithm for backpropagation from this simple definition of gradient descent.



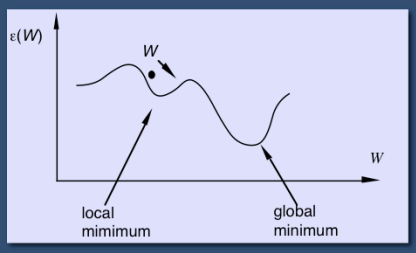
The error function ε must be differentiable with respect to the weights.

Too large a ρ (learning rate, or step size) can cause oscillations.

Too small a learning rate can drastically slow down the algorithm.

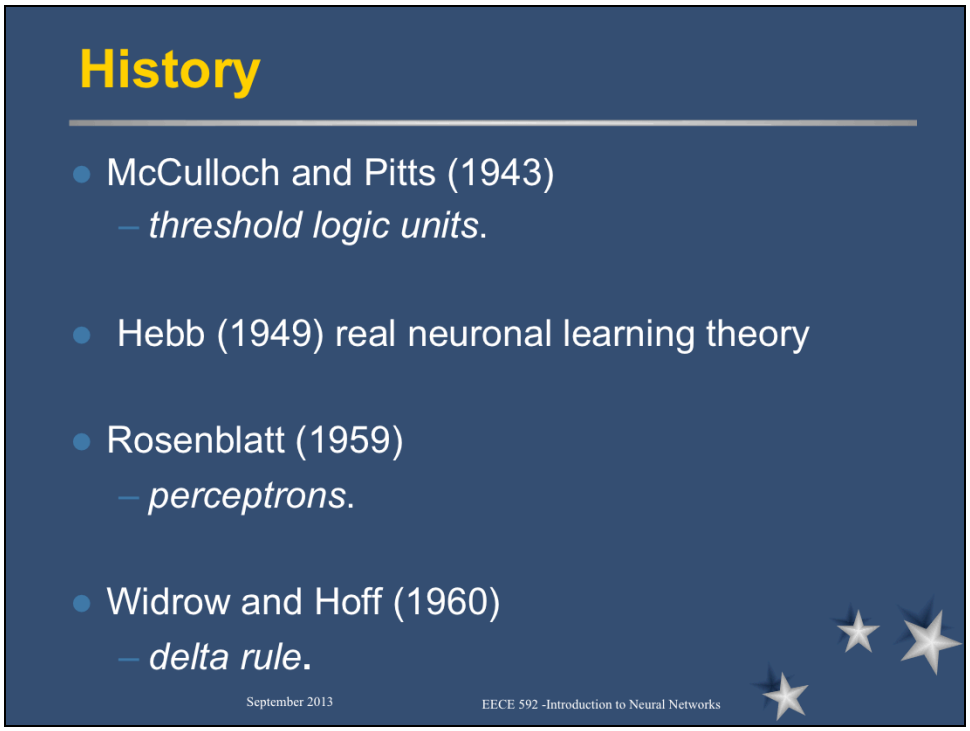
Problems with BP cont.

- Local Minima



The algorithm is not guaranteed to find a global minimum error. In fact, once a minimum has been reached it is very difficult to determine whether it is a global minimum or not!

Although often cited as an issue regarding gradient descent based learning approaches, it should be noted that to be truly problematic, the minima must be along all dimensions simultaneously. Consider a "gutter" for example. It is a minima along its cross-section but not along it lengths. This is what allows it to carry water from one point to another lower.

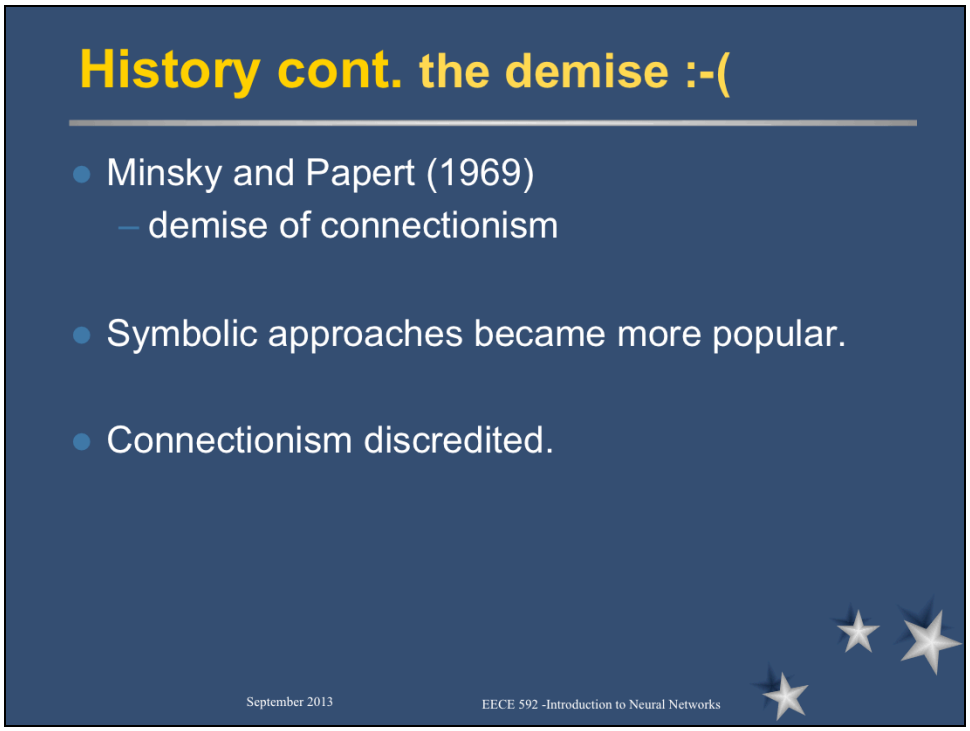


As already mentioned, one of the earliest pieces of work related to neural networks was that by McCulloch and Pitts (1943) who proposed the notion of *threshold logic units*. Effectively simple computing units.

Hebb (1949) suggested that real neuronal learning took place according to a simple rule. Briefly described, the rule stated that if two cells A & B are simultaneously active, then increase the weight of any connection between them.

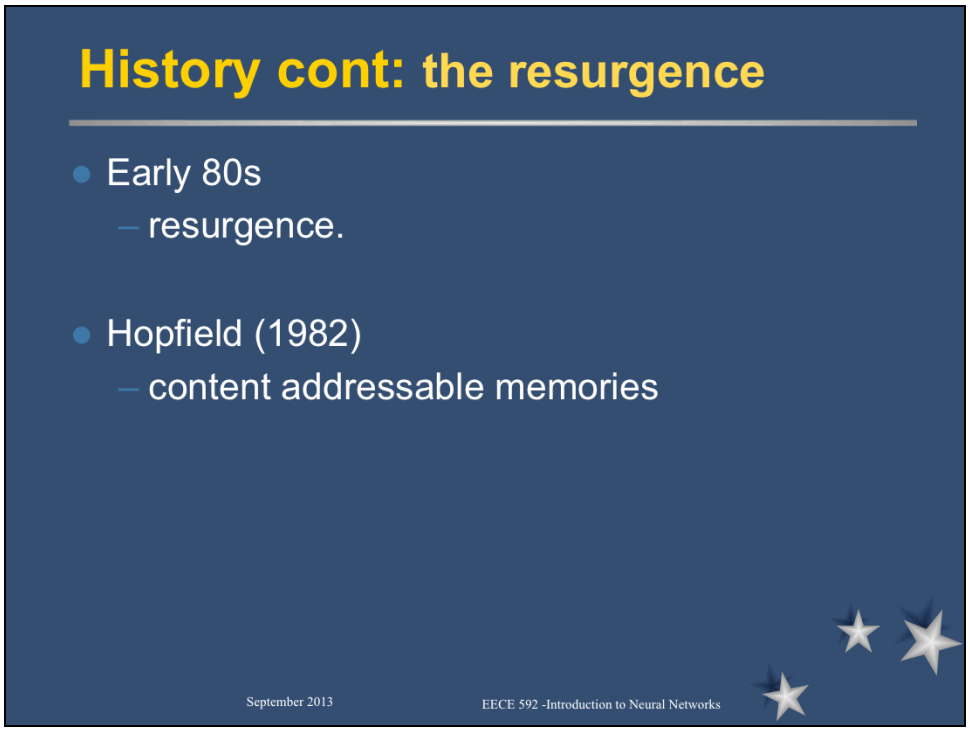
Rosenblatt (1959) devised one of the first algorithms for learning and updating weights in connectionist models. The algorithm was capable of updating weights in simple single layer models known as *perceptrons*.

Widrow and Hoff (1960) developed a variation of Rosenblatts perceptron learning algorithm now known as the *delta rule*.

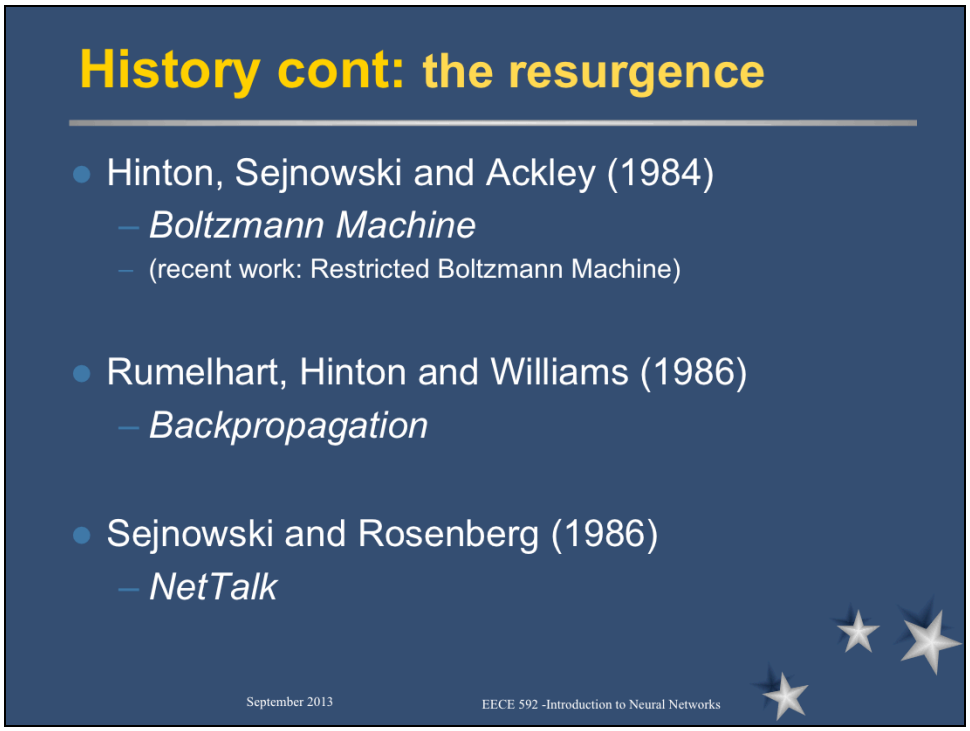


Connectionism, another term for the study of neural networks, became markedly unpopular after a publication by Minsky and Papert (1969). In their work, they rightfully pointed out severe theoretical limitations in Rosenblatts approach.

Not soon after, symbolic approaches (e.g. Winston (1975)) became more and more popular. As did rule based approaches to intelligence. Perceptron learning and neural network models were discredited by all but a few researchers. Academically, choosing to continue research in neural networks was regarded as a career limiting move!



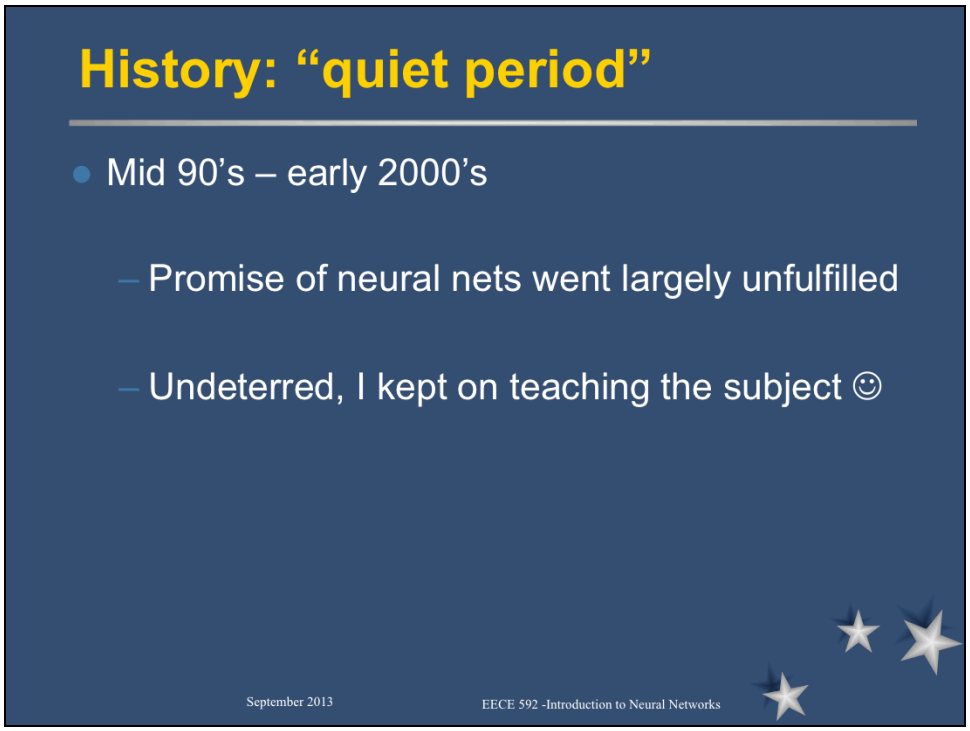
The early 80s however saw a resurgence sparked off by the big names in neural nets. (e.g. Hopfield (1982) talked about content addressable memories)



Hinton, Sejnowski and Ackley (1984) developed the *Boltzmann Machine* which overcame the limitations of learning in single layer models as pointed out by Minsky and Papert. Work continues on a simplified version, called the Restricted Boltzmann Machine. See http://www.cs.toronto.edu/~hinton/

Rumelhart, Hinton and Williams (1986) re-discovered and popularized the backpropagation learning algorithm.

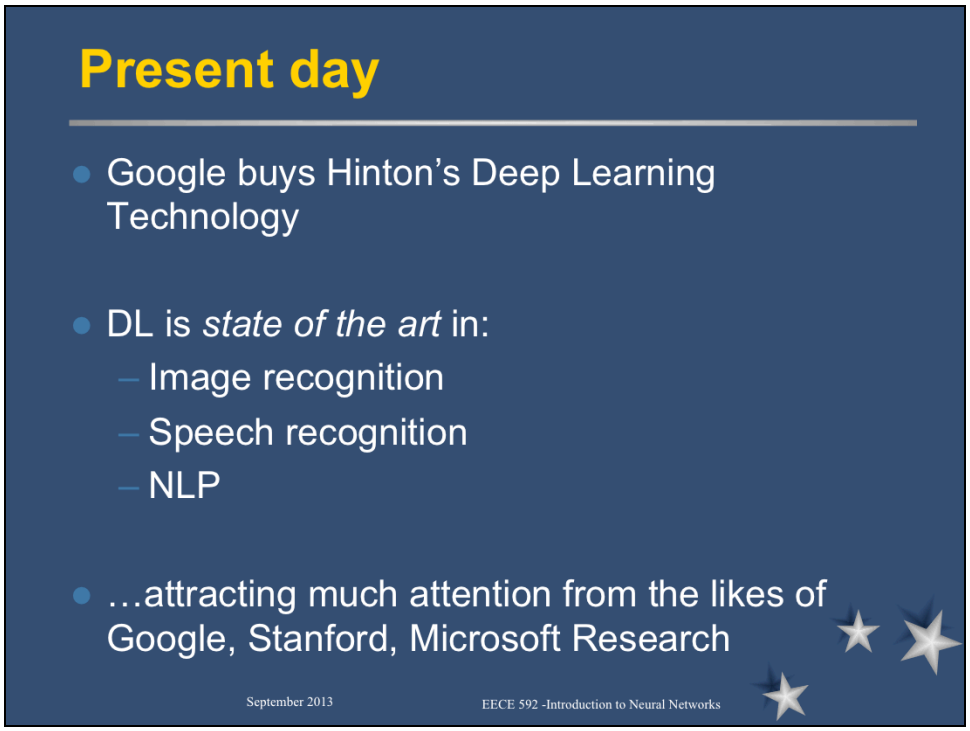
Sejnowski and Rosenberg (1986) showed how it could be used to teach a connectionist model to pronounce English text. The system known as NetTalk is a landmark in the resurgence of neural network models.



Although the initial excitement of the late 80s, early 90s has worn off, there is still much research continuing in the field of neural networks. The video gaming industry has shown interest in creating intelligent artificial opponents. Neural networks see good application in robot control theory, speech and audio processing applications and offer novel solutions to pattern recognition problems.

But wait, there’s more……

Around 2006, some interesting ideas were emerging. Hinton's Restricted Boltzmann Machine demonstrated how a Boltzmann Machine could generate useful internal representations, if connections are restricted to give a layered network topology. The idea of convolution (originally introduced by Yann LeCun) and sparse auto-encoders lead to the development of unsupervised learning approaches that could be used to train many layers of a multi-layer perceptron in a biologically "optimistic" way. Particularly, the visualization of the internal layers after training, showed interesting parallels with the structure of the visual cortex as discovered by Hubel and Wiesel back in the late 60s.



Fast forward to the present day (2013). Hinton used a Deep Neural Net to classify high resolution images in the 2010 ImageNet competition. Here are some of the details:

- Network
    - 650k neurons
    - 60 million weights
    - 9+ layers
- Data
    - 1.2 million images / 1000 classes
- CPU
    - 2 days of training on 2 GTX 580 3GB GPUs
- Results
    - Top 1% error rate = 37.5%
    - Top 5% error rate = 17.5%
        - This is signficantly better than state of the art!

Published in Krizhevsky, Alex, Ilya Sutskever, and Geoff Hinton. "Imagenet classification with deep convolutional neural networks." *Advances in Neural Information Processing Systems 25*. 2012.