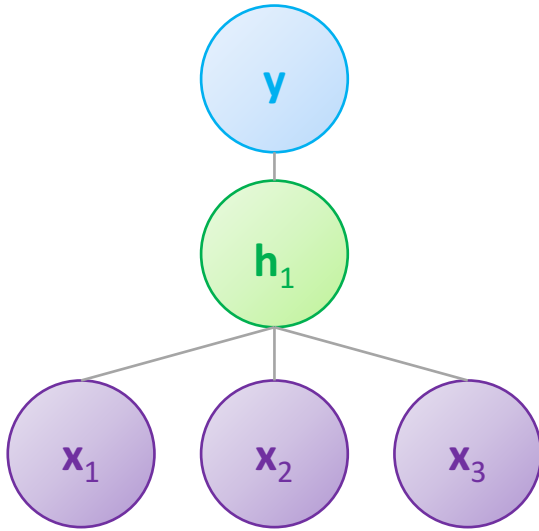


Artificial neural networks

PROS	CONS
Few assumptions	Narrowly accepted
Can sometimes handle missing values	Narrowly understood
Can accurately model extremely nonlinear phenomena	Difficult to interpret
Tend to excel at image, sound, and other pattern recognition tasks	Tendency to overfit

Wait a second ... what is an artificial neural network?

Many neural networks are just sophisticated **regression models**

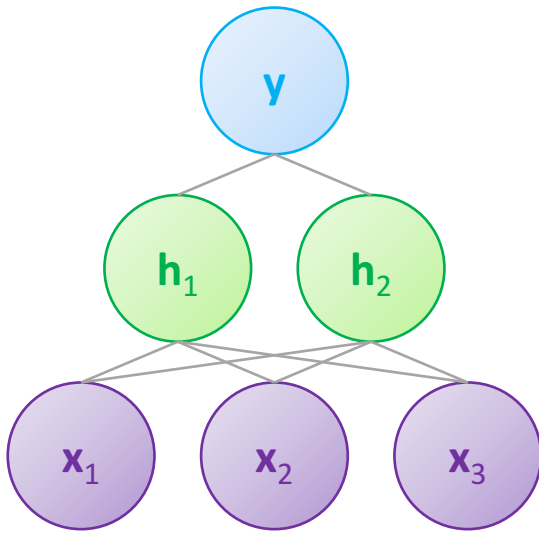


$$y = \tanh(w_0 + w_1x_1 + w_2x_2 + w_3x_3)$$

Neural networks are similar to some biological mechanisms in the brain

Wait a second ... what is an artificial neural network?

Many neural networks are just sophisticated **regression models**

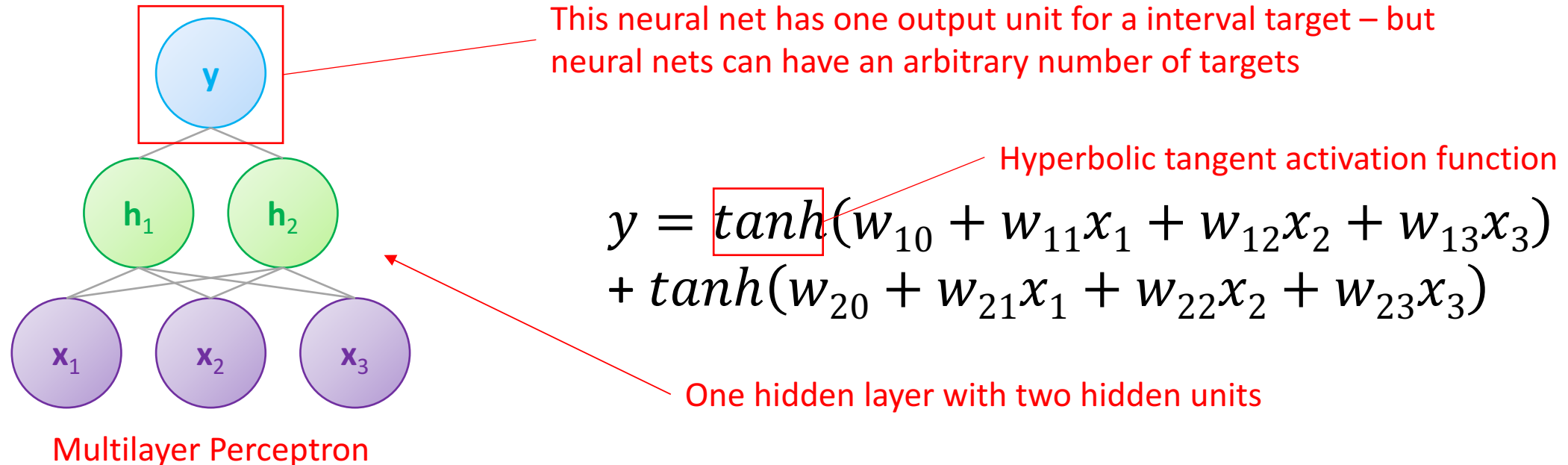


$$y = \tanh(w_{10} + w_{11}x_1 + w_{12}x_2 + w_{13}x_3) + \tanh(w_{20} + w_{21}x_1 + w_{22}x_2 + w_{23}x_3)$$

Neural networks are similar to some biological mechanisms in the brain

Wait a second ... what is an artificial neural network?

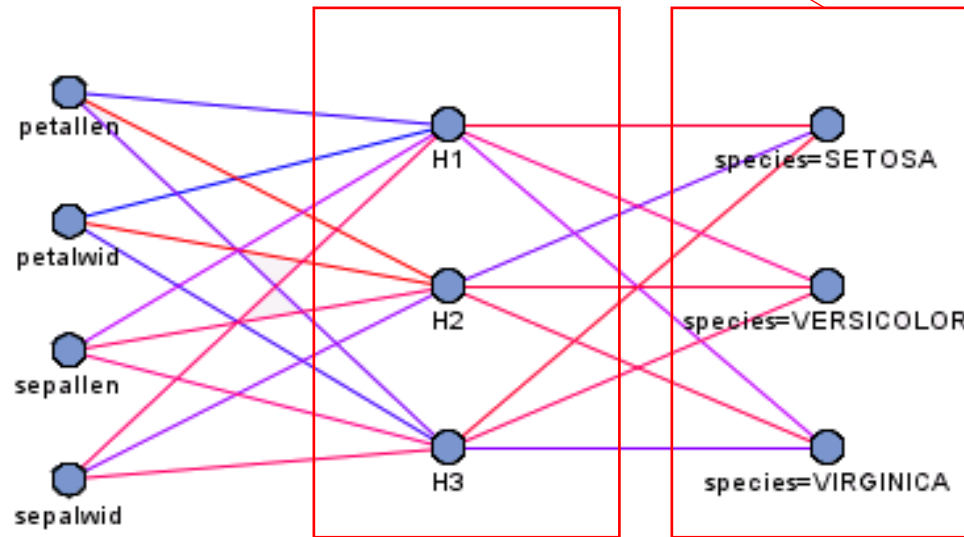
Many neural networks are just sophisticated **regression models**



Neural networks are similar to some biological mechanisms in the brain

Another simple ANN for iris species classification

This neural net has three output units for a single categorical target (but neural nets can have an arbitrary number of targets)

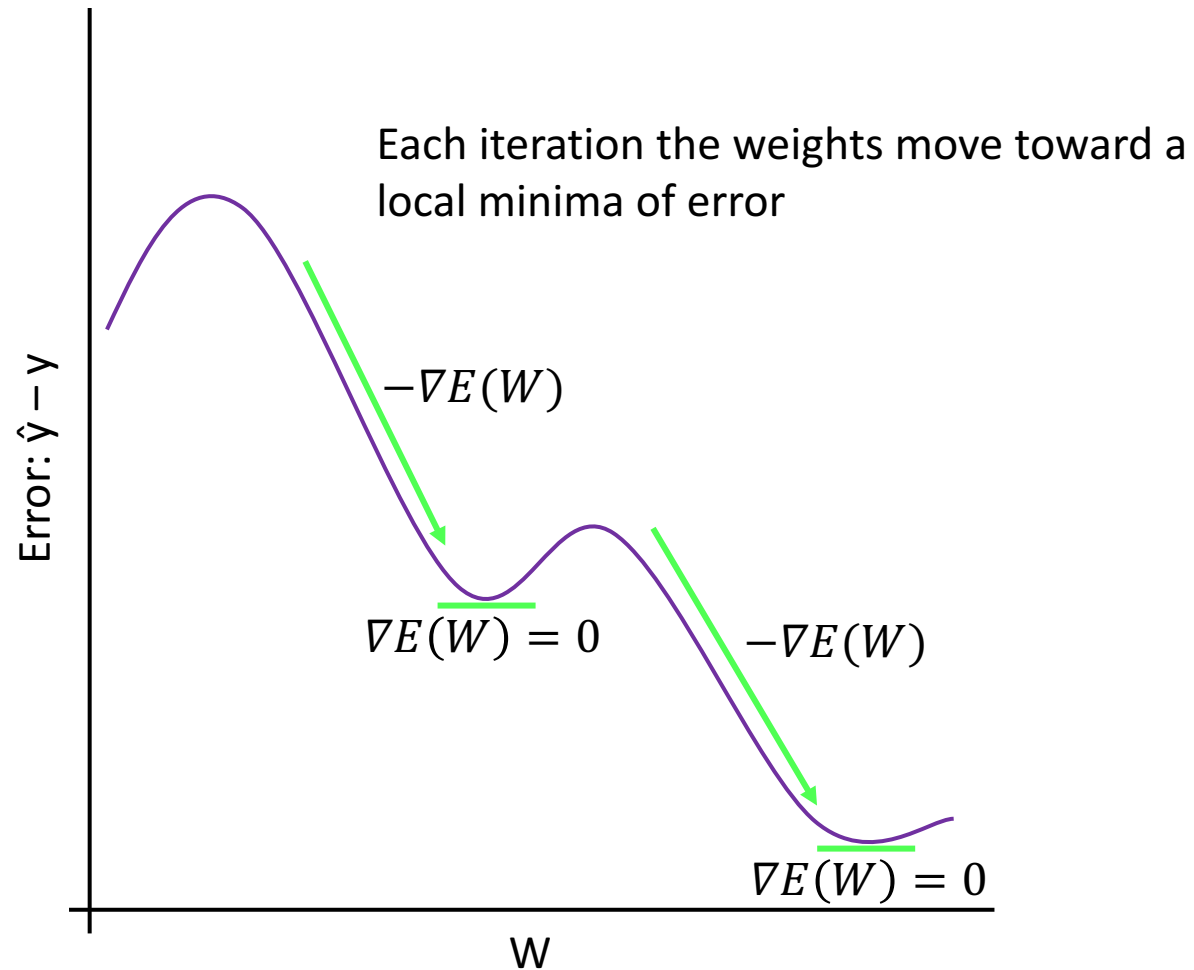


Weights from all layers tend to be meaningless – they are simply the weights that minimized the prediction error on this data; different weights can lead to the same error rate!

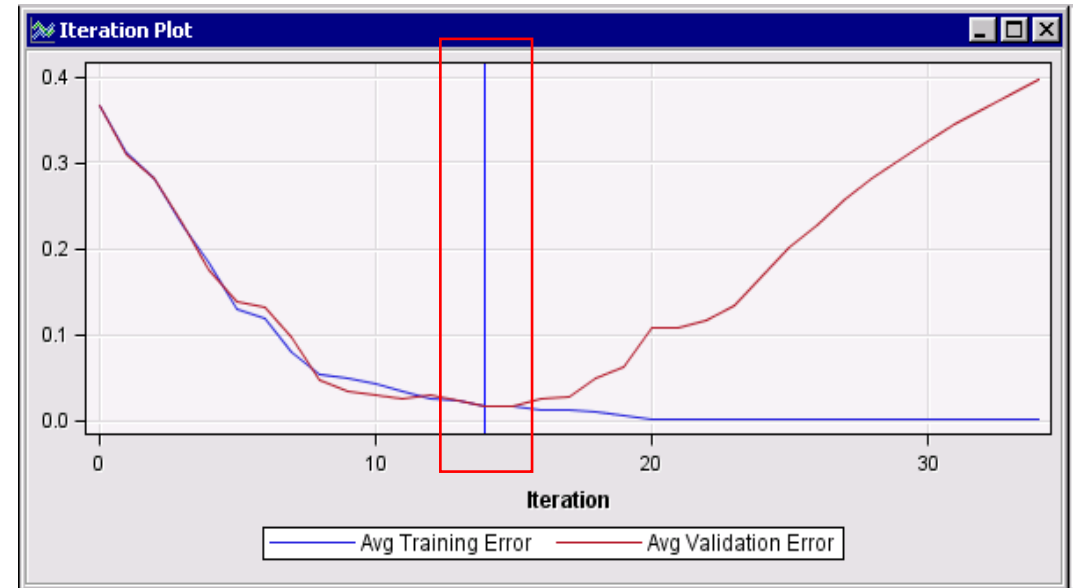
Multilayer Perceptron

This neural net has three hidden units in one hidden layer

Training neural networks: gradient descent



This neural network appears to reach a minimum in validation error after about 14 iterations



Neural networks: predictions

Petal Length (mm)	Petal Width (mm)	Sepal Length (mm)	Sepal Width (mm)
20	15	14	18

* STANDARDIZE INPUTS;

S_petallen = $-2.13 + 0.05 * \text{petallen}$;

S_petalwid = $-1.57 + 0.13 * \text{petalwid}$;

S_sepallen = $-7.05 + 0.12 * \text{sepallen}$;

S_sepalwid = $-7.04 + 0.23 * \text{sepalwid}$;

* APPLY HIDDEN LAYER WEIGHTS;

H11 = $-1.49 * S_petallen + 7.94 * S_petalwid + 1.94 * S_sepallen + 1.11 * S_sepalwid$;

H11 = $-6.02 + H11$;

H11 = $\text{TANH}(H11)$;

H12 = $5.05 * S_petallen - 2.23 * S_petalwid + 8.34 * S_sepallen - 9.36 * S_sepalwid$;

H12 = $0.64 + H12$;

H12 = $\text{TANH}(H12)$;

H13 = $24.18 * S_petallen + 5.00 * S_petalwid - 8.98 * S_sepallen + 0.79 * S_sepalwid$;

H13 = $-11.28 + H13$;

H13 = $\text{TANH}(H13)$;

* APPLY OUTPUT LAYER WEIGHTS;

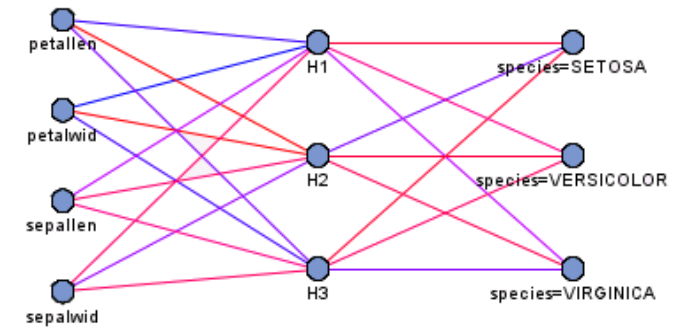
P_speciesvirginica = $9.60 * H11 + 30.93 * H12 + 34.21 * H13$;

P_speciesvirginica = $-5.42 + P_speciesvirginica$;

P_speciesversicolor = $-8.71 * H11 + 15.72 * H12 + 14.00 * H13$;

P_speciesversicolor = $10.28 + P_speciesversicolor$;

P_speciessetosa = 0;



Neural networks: predictions

Petal Length (mm)	Petal Width (mm)	Sepal Length (mm)	Sepal Width (mm)
20	15	14	18

* STANDARDIZE INPUTS;

S_petallen = $-2.13 + 0.05 * \text{petallen}$; (= -1.13)

S_petalwid = $-1.57 + 0.13 * \text{petalwid}$; (= 0.38)

S_sepallen = $-7.05 + 0.12 * \text{sepallen}$; (= -5.37)

S_sepalwid = $-7.04 + 0.23 * \text{sepalwid}$; (= -2.9)

Step 1

* APPLY HIDDEN LAYER WEIGHTS;

H11 = $-1.49 * S_petallen + 7.94 * S_petalwid + 1.94 * S_sepallen + 1.11 * S_sepalwid$;

H11 = $-6.02 + H11$;

H11 = $\text{TANH}(H11)$; (= -1)

H12 = $5.05 * S_petallen - 2.23 * S_petalwid + 8.34 * S_sepallen - 9.36 * S_sepalwid$;

H12 = $0.64 + H12$;

H12 = $\text{TANH}(H12)$; (= -1)

H13 = $24.18 * S_petallen + 5.00 * S_petalwid - 8.98 * S_sepallen + 0.79 * S_sepalwid$;

H13 = $-11.28 + H13$;

H13 = $\text{TANH}(H13)$; (=0.99)

Step 2

* APPLY OUTPUT LAYER WEIGHTS;

P_speciesvirginica = $9.60 * H11 + 30.93 * H12 + 34.21 * H13$;

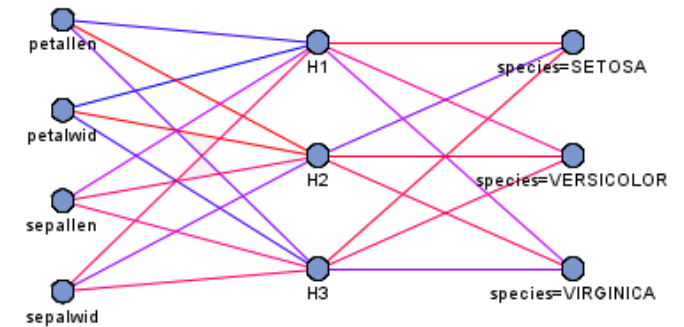
P_speciesvirginica = $-5.42 + P_speciesvirginica$; (= -11.74)

P_speciesversicolor = $-8.71 * H11 + 15.72 * H12 + 14.00 * H13$;

P_speciesversicolor = $10.28 + P_speciesversicolor$; (= 17.27)

P_speciessetosa = 0; (=0)

Versicolor has the highest unnormalized posterior probability



This neural network will never classify anything as Setosa – does this make sense?

Unsupervised training

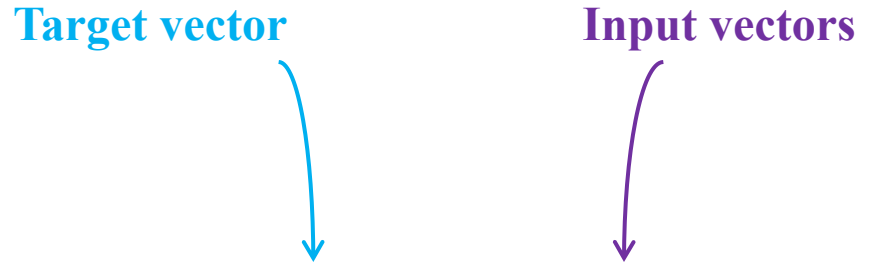
How can a neural network be unsupervised?

- Usually you try to predict a target vector \mathbf{y} from input vectors \mathbf{x}
- In unsupervised training, you try to predict \mathbf{x} from \mathbf{x}

Unsupervised training

Target vector

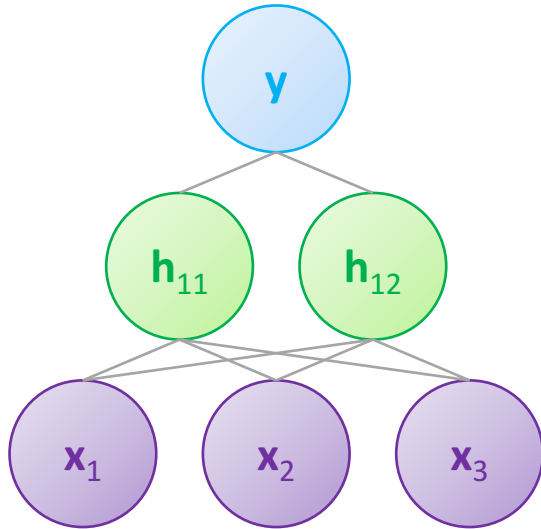
Input vectors



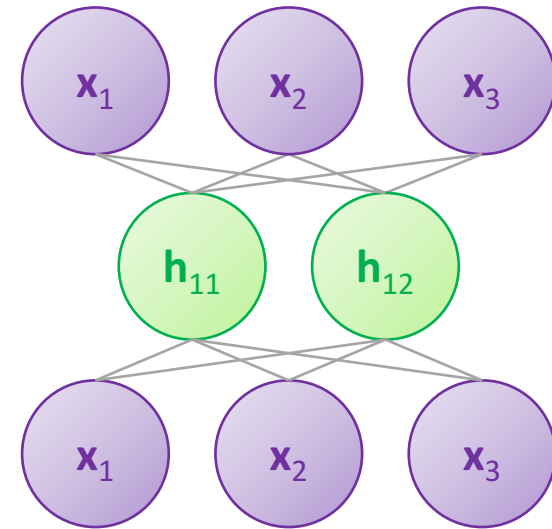
y	x_1	x_2	x_3
1	2.54	1.65	0.02
0	1.14	0.70	0.82
1	0.99	0.51	2.11
\vdots	\vdots	\vdots	\vdots

Unsupervised training

Supervised Neural Network



Unsupervised Neural Network



(Known as an autoencoder)

Unsupervised training

Why is unsupervised training of neural networks useful?

- Feature extraction

Is it trivial to learn \mathbf{x} from \mathbf{x} ?

- Sometimes it is, but the neural network simply learns to duplicate the training data instead of learning **generalizable concepts** from the training data
- To avoid this problem, you can use **L2 regularization** which is equivalent to corrupting the training set by adding Gaussian noise
- A **denoising autoencoder** is a single-hidden layer unsupervised neural network with L2 regularization or Gaussian noise added to the training data

What is deep learning?

“Representation learning is a set of methods that allows a machine to be fed with raw data and to automatically discover the representations needed for detection or classification. Deep-learning methods are representation-learning methods with multiple levels of representation, obtained by composing simple but non-linear modules that each transform the representation at one level (starting with the raw input) into a representation at a higher, slightly more abstract level ... For classification tasks, higher layers of representation amplify aspects of the input that are important for discrimination and suppress irrelevant variations.”

-- Yann LeCun, Yoshua Bengio, Geoffrey Hinton, 2015

Why is deep important?

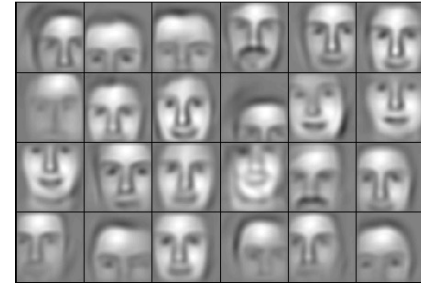
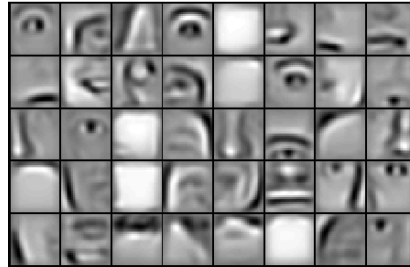
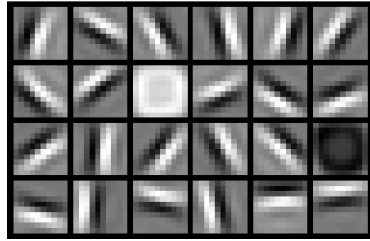
Neural networks with many layers achieve state-of-the-art results in pattern recognition

Also:

- The bottom layer(s) of a deep network learn **optimal features**
- More layers are usually **more efficient** than more neurons for attaining a given level of accuracy
- Multiple layers give information about the training data at **multiple scales (*representation learning*)**

Why is deep important?

Feature extraction at multiple scales:

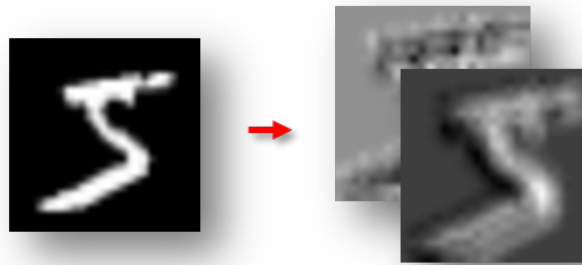


Stacked layers: convolutional neural networks

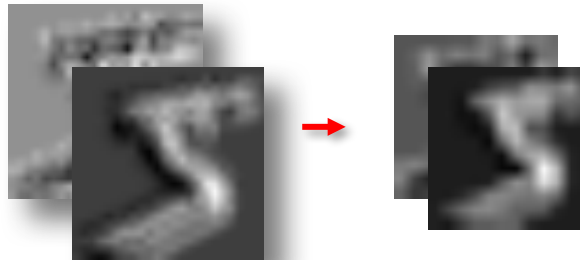
- Built by stacking layers composed of image convolution and down-sampling (pooling) followed by conventional hidden layers
- Typically used for supervised pattern recognition tasks
- Image convolution and down-sampling reduce the number of hidden units needed in upper layers of the network while also extracting robust features from the training data

Convolution and down-sampling

- Convolution applies a filter to a neighborhood of pixels



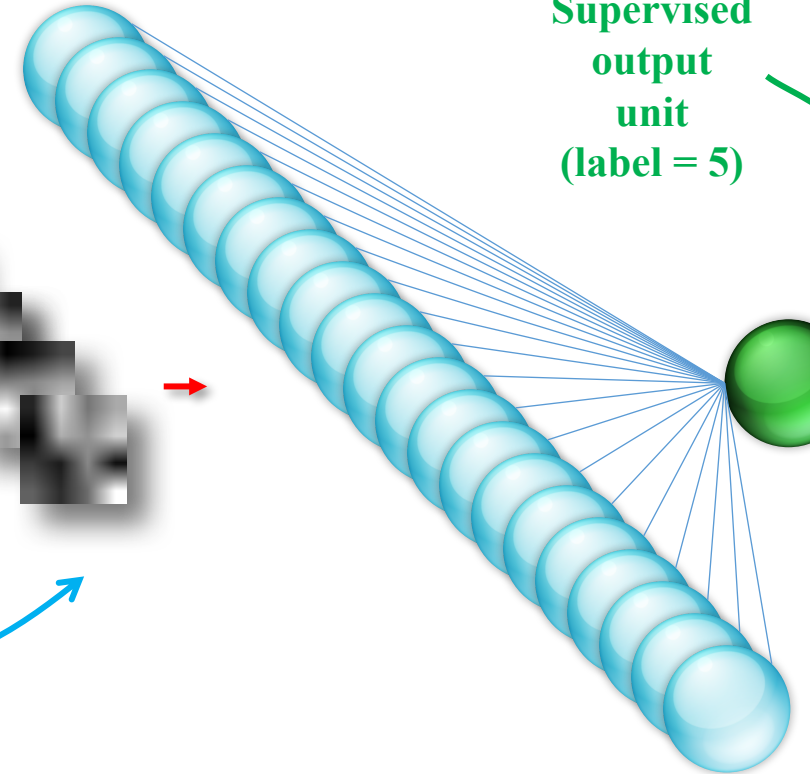
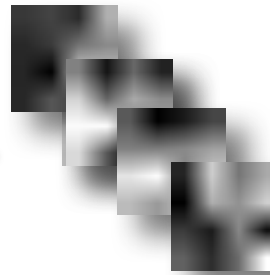
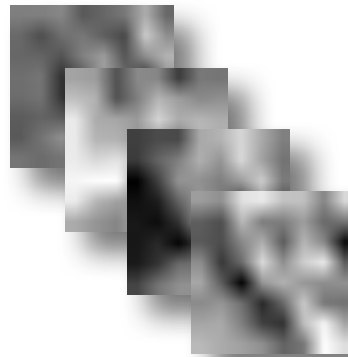
- Down-sampling summarizes a group of pixels



The number of convolutional filters at each layer is a hyperparameter of the network. Filter values are optimized along with hidden unit weights during supervised training.

All images of each input example are flattened into a single vector and fed to an Multi-layer perceptron classifier.

Input vectors representing pixels



Supervised output unit (label = 5)

Down-sampling reduces computational complexity for upper layers and provides a form of scale and translation invariance.

1st convolutional layer

1st down-sampling layer

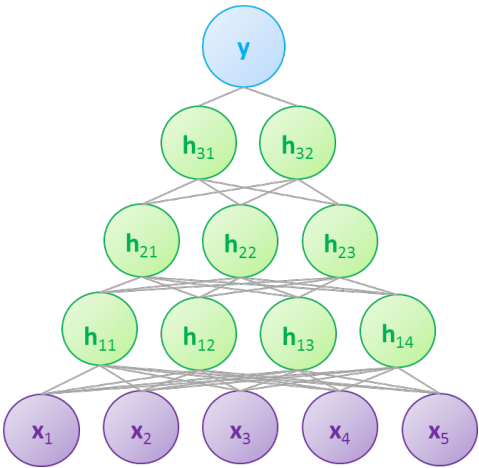
2nd convolutional layer

2nd down-sampling layer

Fully connected MLP classifier

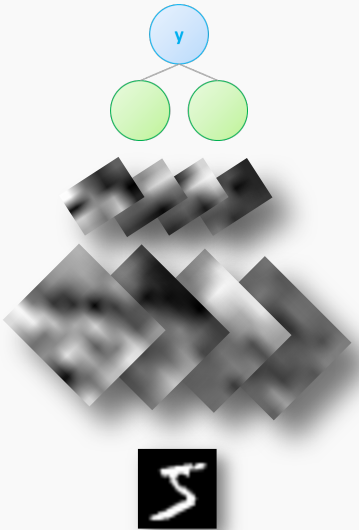
Deep Supervised Network

Conventional Hidden
Layers



Deep Neural Network

Convolution, Down-
sampling, conventional
Hidden layers



Convolutional neural network