Computational Data Analysis

Machine Learning

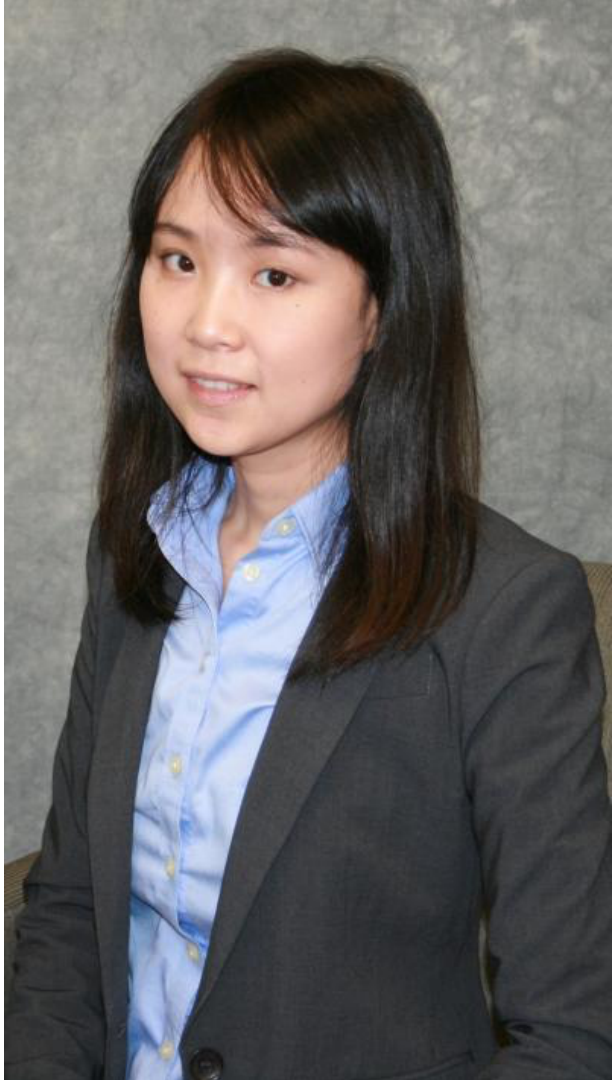
Yao Xie, Ph.D.

Associate Professor

Harold R. and Mary Anne Nash Early Career Professor

H. Milton Stewart School of Industrial and Systems Engineering

Introduction to Neural Networks

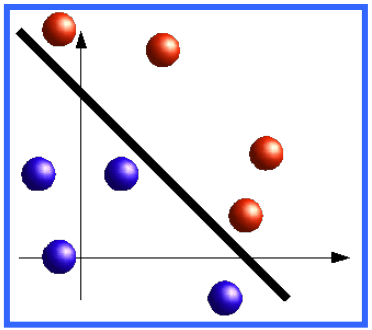


Main approaches to design classifiers

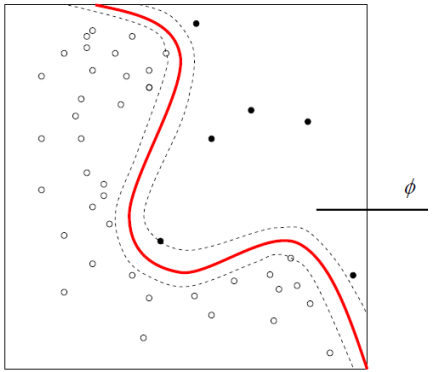
- Bayes rule, use simplifying assumption for $p(x|y=1)$
  - Assume $p(x|y=1)$ is Gaussian
  - Assume $p(x|y=1)$ is fully factorized
- Use geometric intuitions
  - k-nearest neighbor classifier
  - Support vector machine
- Directly go for the decision boundary $\mathrm{h(x)} = -\ln \frac{q_i(x)}{q_j(x)}$
  - Logistic regression
  - Neural networks

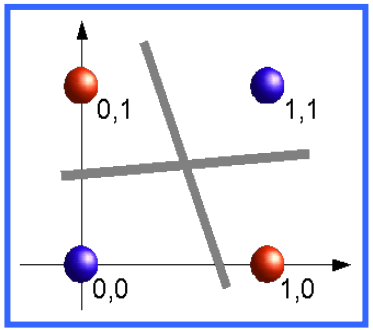
Learning nonlinear decision boundary

Linearly separable

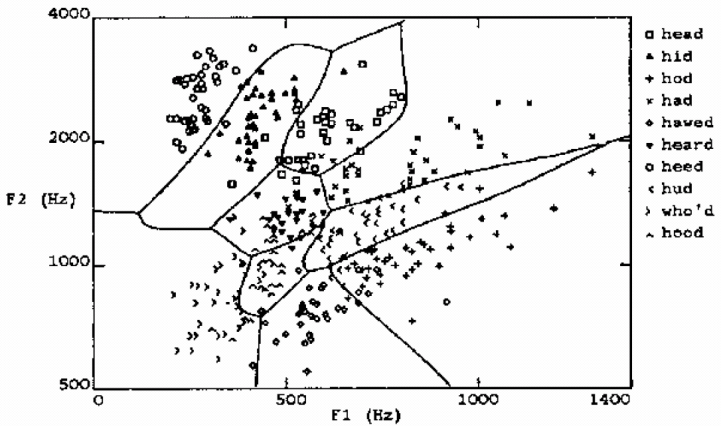


Nonlinearly separable





The XOR gate

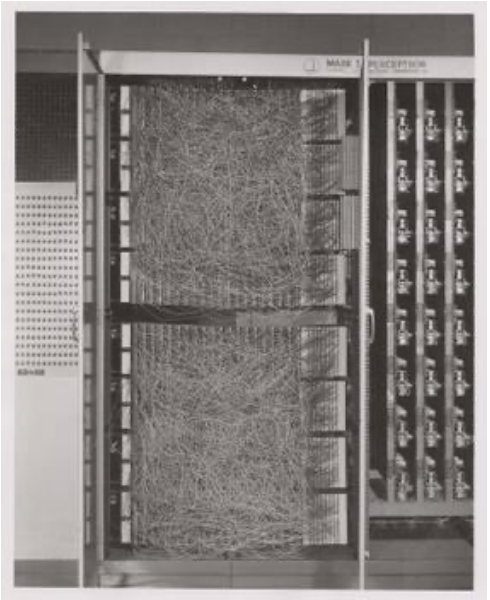


Speech recognition

Early artificial intelligence

- The perceptron algorithm was first invented in 1958 at the Cornell Aeronautical Lab

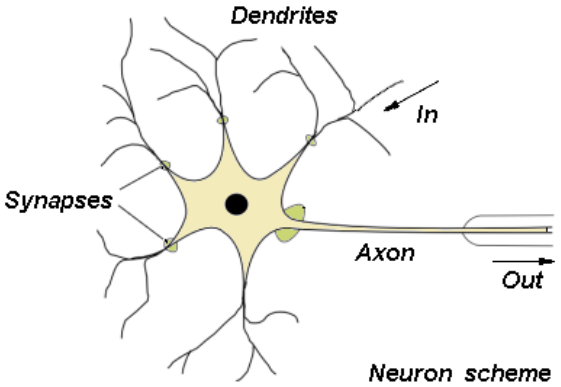
- This machine was designed for image recognition: it had an array of 400 photocells, randomly connected to the "neurons".

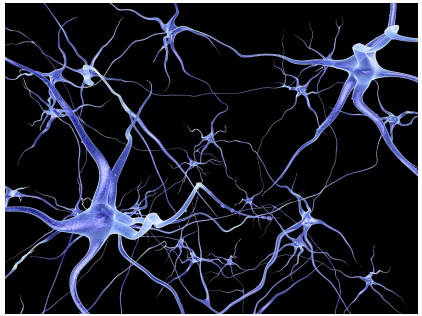


Mark I Perceptron machine, the first implementation of the perceptron algorithm. It was connected to a camera with 20×20 cadmium sulfide photocells to make a 400-pixel image. The main visible feature is a patch panel that set different combinations of input features. To the right, arrays of potentiometers that implemented the adaptive weights.

Perceptron

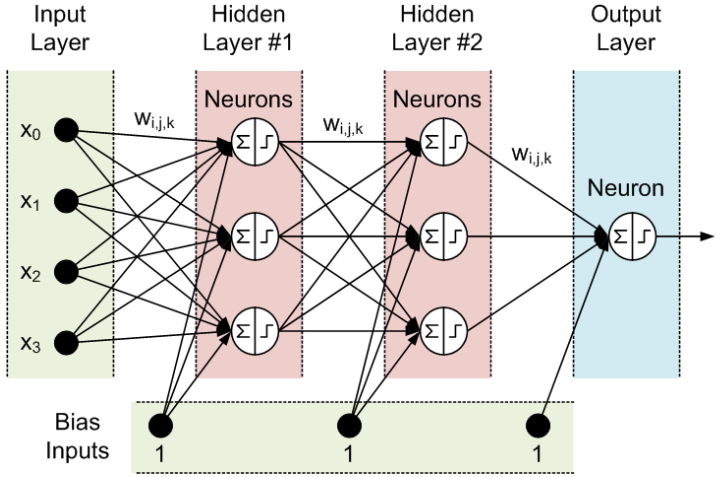
- From biological neurons to artificial neurons (perceptron)



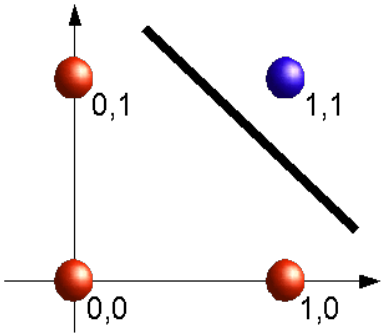


Human brain: Number of neurons ~ 10^{10}

- Artificial neural networks (ANN)
  - Many neuron-like threshold switching units
  - Weighted interconnections among units at layers (feed-forward)



Perceptron



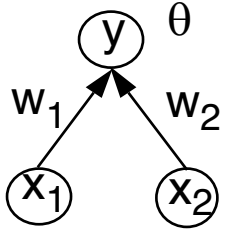
NAND

x	y	Z (color)
0	0	**0**
0	1	**0**
1	0	**0**
1	1	**1**

Threshold $\theta = 0.5$

some possible values for w_1 and w_2

w_1	w_2
0.20	0.35
0.20	0.40
0.25	0.30
0.40	0.20



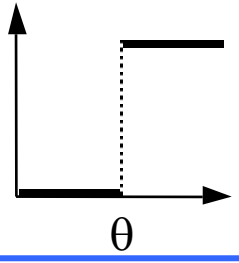
$f(x_1w_1 + x_2w_2) = y$

$f(0w_1 + 0w_2) = 1$

$f(0w_1 + 1w_2) = 1$

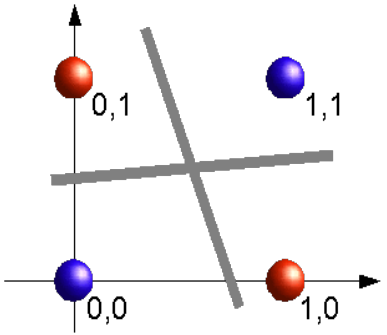
$f(1w_1 + 0w_2) = 1$

$f(1w_1 + 1w_2) = 0$



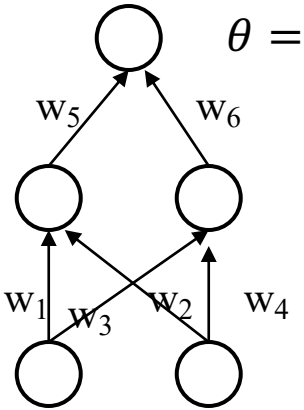
$$f(u) = \begin{cases} 1, \text{ for } u > \theta \\ 0, \text{ for } u \leq \theta \end{cases}$$

Connecting perceptron can give more complex decision boundary



NAND

x	y	Z (color)
0	0	**0**
0	1	**1**
1	0	**1**
1	1	**0**



$\theta = 0.5$ *for all units*

a possible set of values for $(w_1, w_2, w_3, w_4, w_5, w_6)$:
$(0.6,-0.6,-0.7,0.8,1,1)$

Expressive capability of ANN

- Boolean functions:
  - Every Boolean function can be represented by network with single hidden layer
  - But might require exponential (in number of inputs) hidden units
- Continuous functions:
  - Every bounded continuous function can be approximated with arbitrary small error, by network with one hidden layer [Cybenko 1989; Hornik et al 1989]
  - Any function can be approximated to arbitrary accuracy by a network with two hidden layers [Cybenko 1988].
  - Many recent developments for more general activation functions (e.g. ReLU)

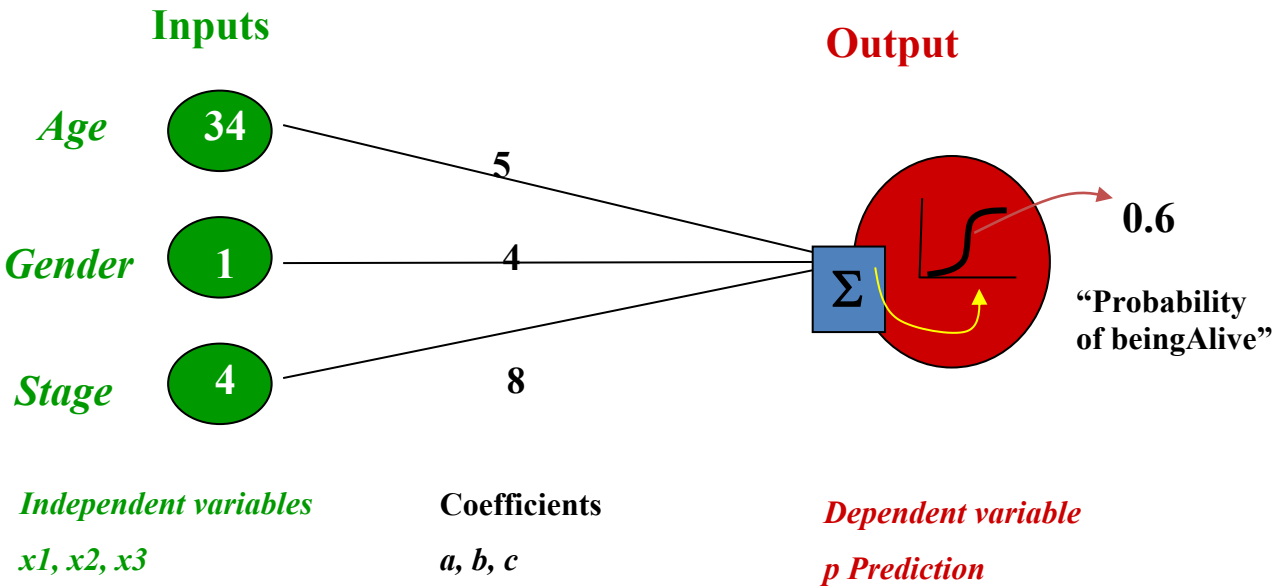
Logistic regression can be viewed as a “single neuron”

Independent variable = input variable

Dependent variable = output variable

Coefficients = “weights” + “bias”

Logistic Regression Model (the sigmoid unit) is a perceptron

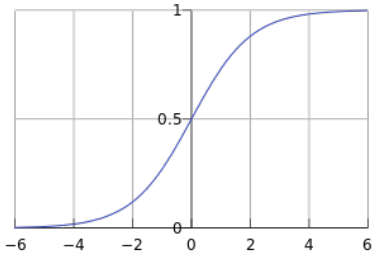


Connection to logistic regression

- Let the original feature be denoted as u (n dimensional vector)
- Note that we can define the augmented feature as $x = [1; u^T]^T$ (a $n+1$ dimensional vector)
- The weight vector $\widetilde{w} = [b, w^T]^T$
- Assume that the posterior distribution $p(y = 1|x)$ take a particular form

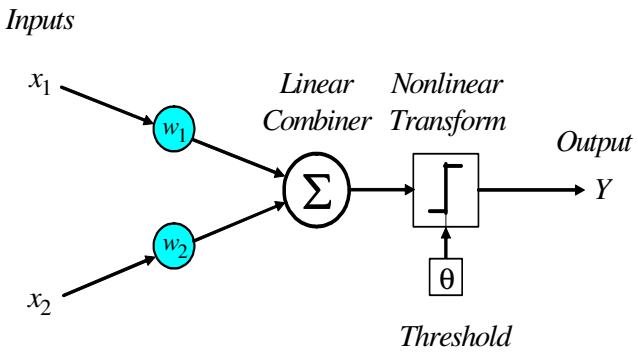
$$p(y = 1|x, \widetilde{w}) = \frac{1}{1 + \exp(-\widetilde{w}^\top x)} = \sigma(\widetilde{w}^\top x)$$

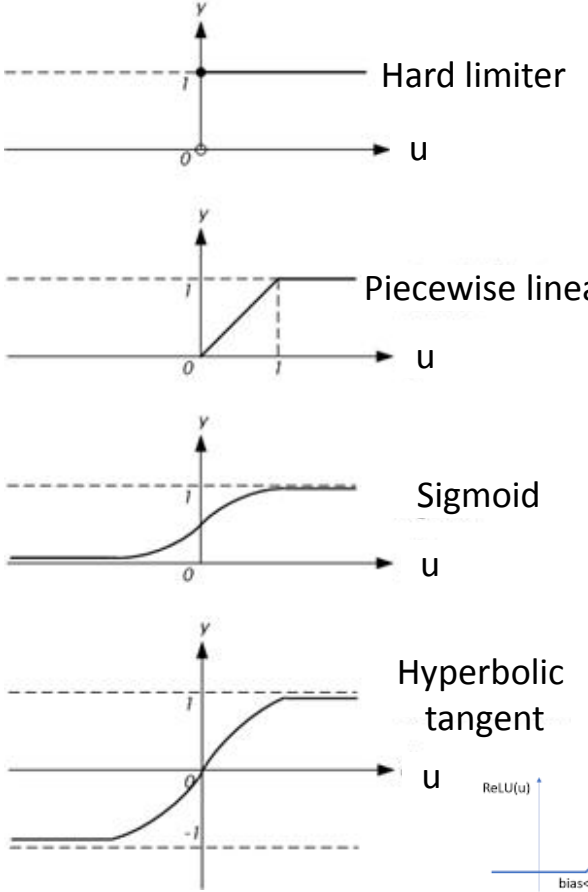
- Logistic function (or sigmoid function) $\sigma(u) = \frac{1}{1+\exp(-u)}$
- Logistic regression can be viewed as one neuron

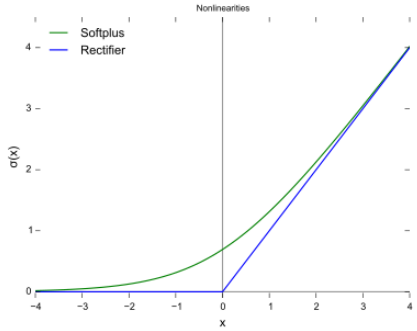


Other nonlinear neurons

- Use different nonlinear transformations $f(u)$
- Before that, perform weighted combination of inputs $u = w^{\top}x$





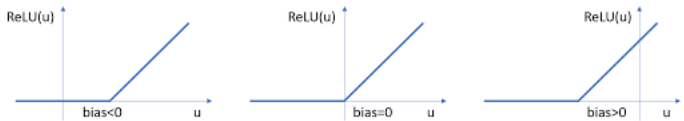


Rectifier linear unit (ReLU)

$$f(u) = \max\{u, 0\}$$

With bias

$$f(u;\theta) = \max\{u + \theta, 0\}$$

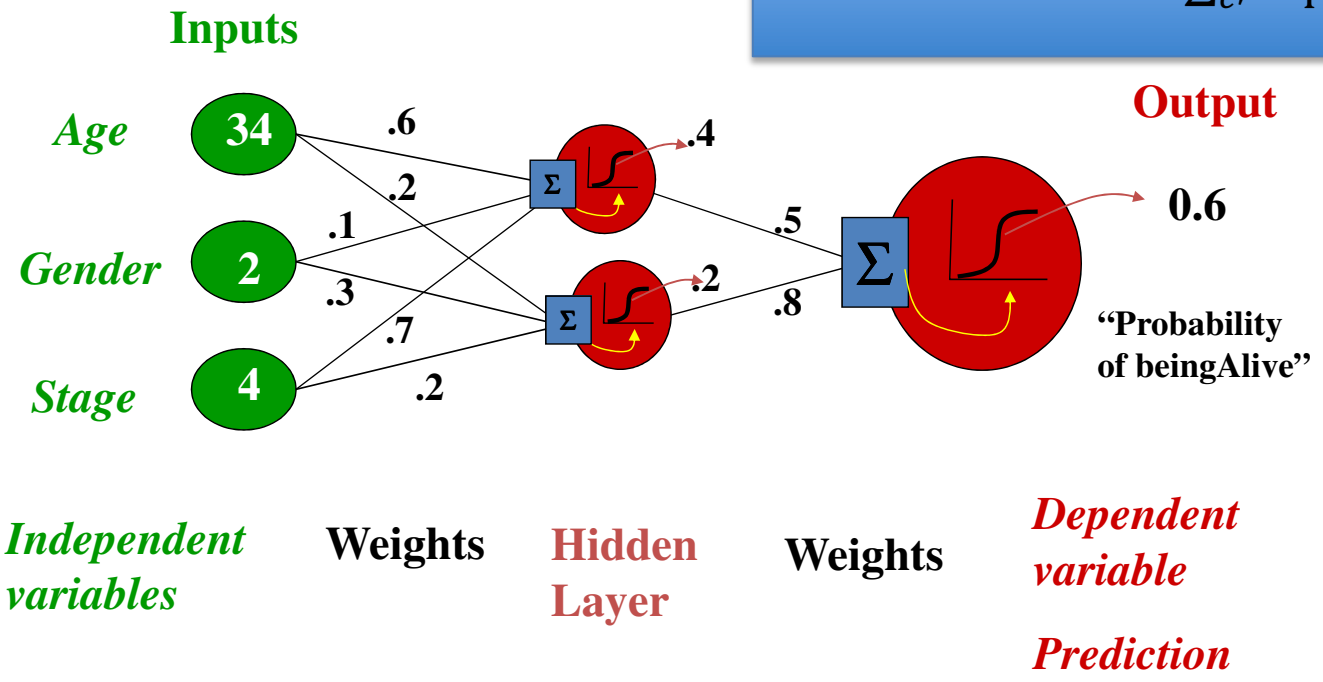


Deep Neural Network Model

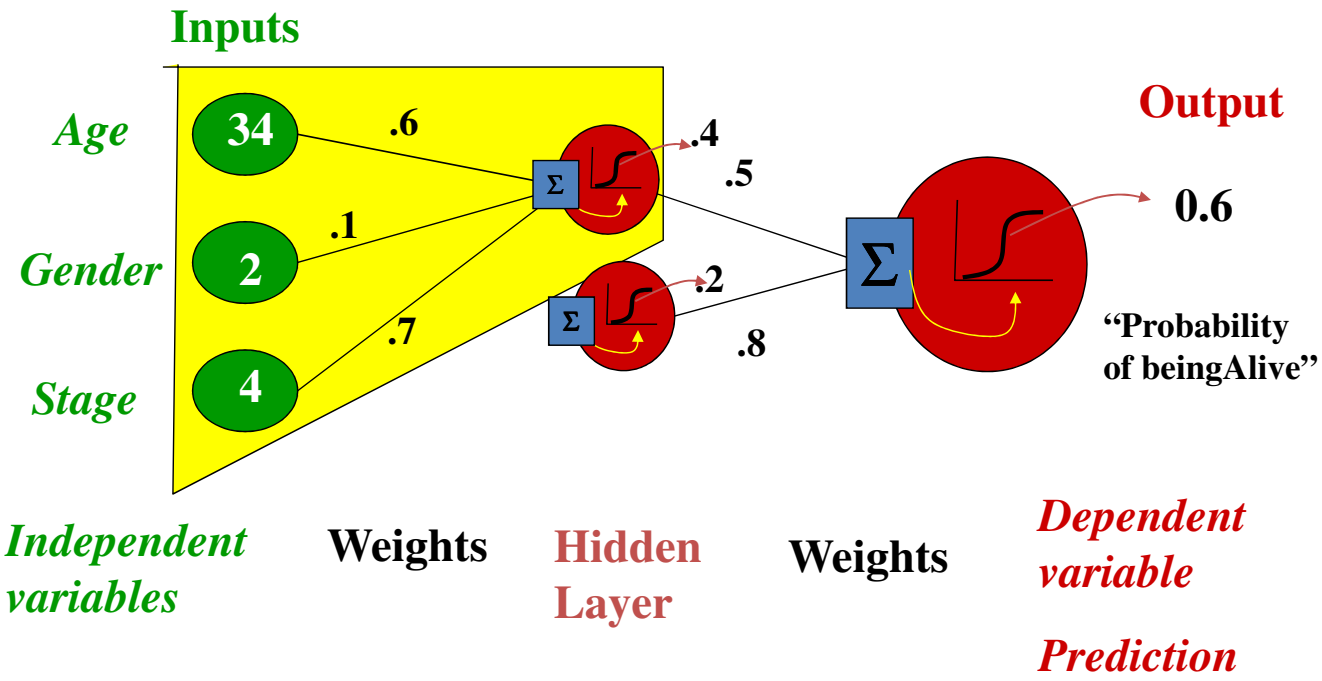
Multiple layers of interconnected perceptrons

The last layer can be multi-class classification such as soft-max function

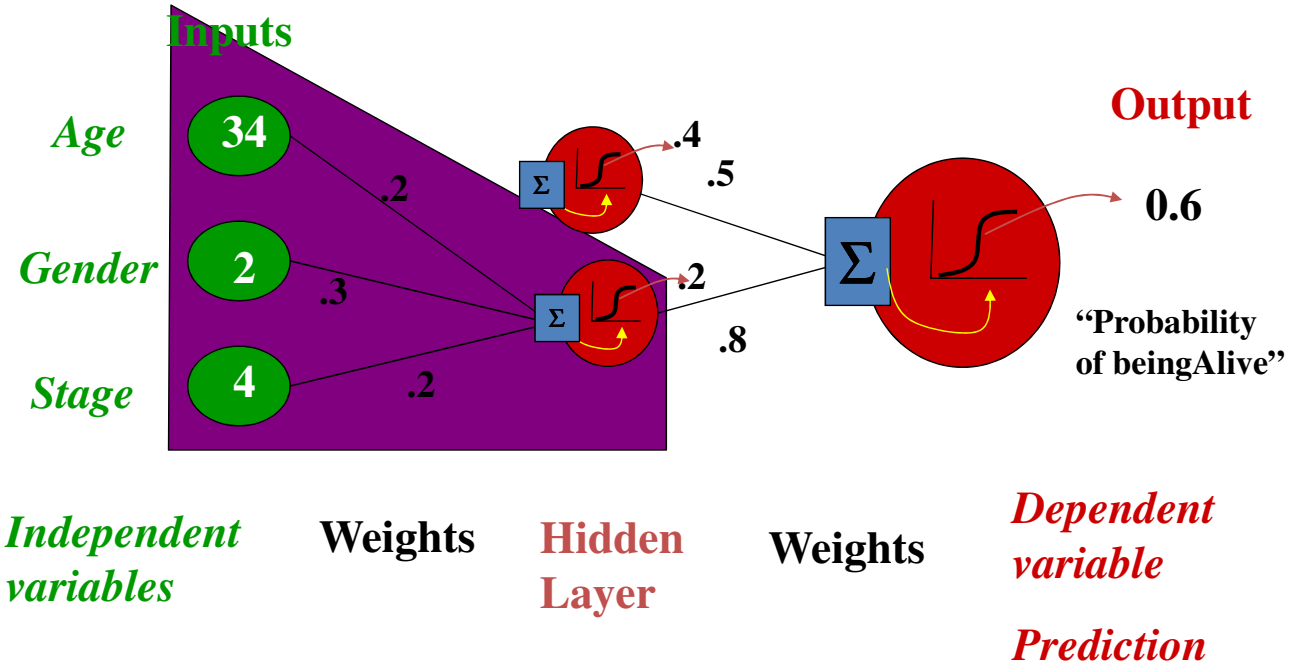
$$P(y^i = c|x^i) = \frac{\exp(\theta_c{}^\top x^i)}{\sum_{c\prime} \exp(\theta_{c\prime}{}^\top x^i)}$$



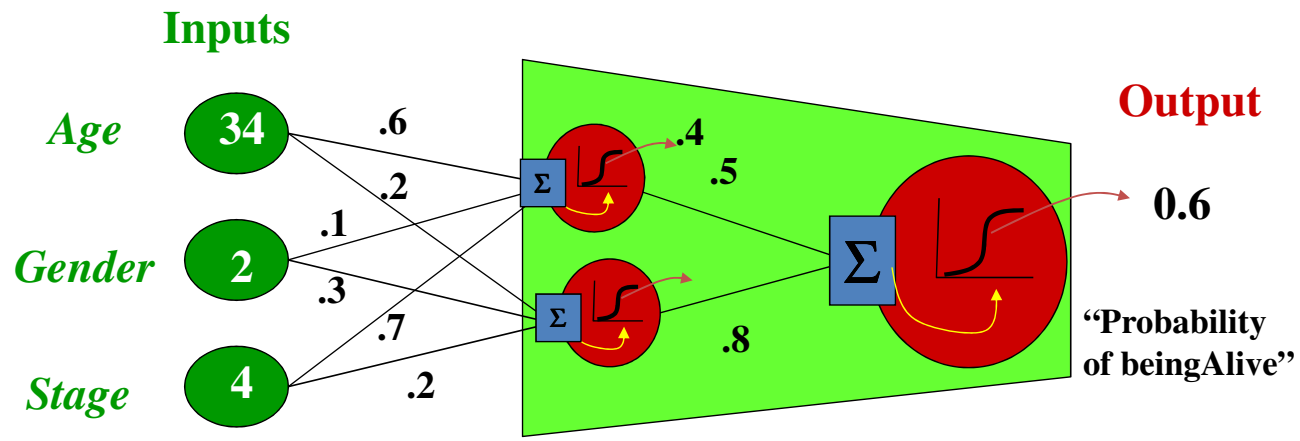
“Combined logistic models”



“Combined logistic models”

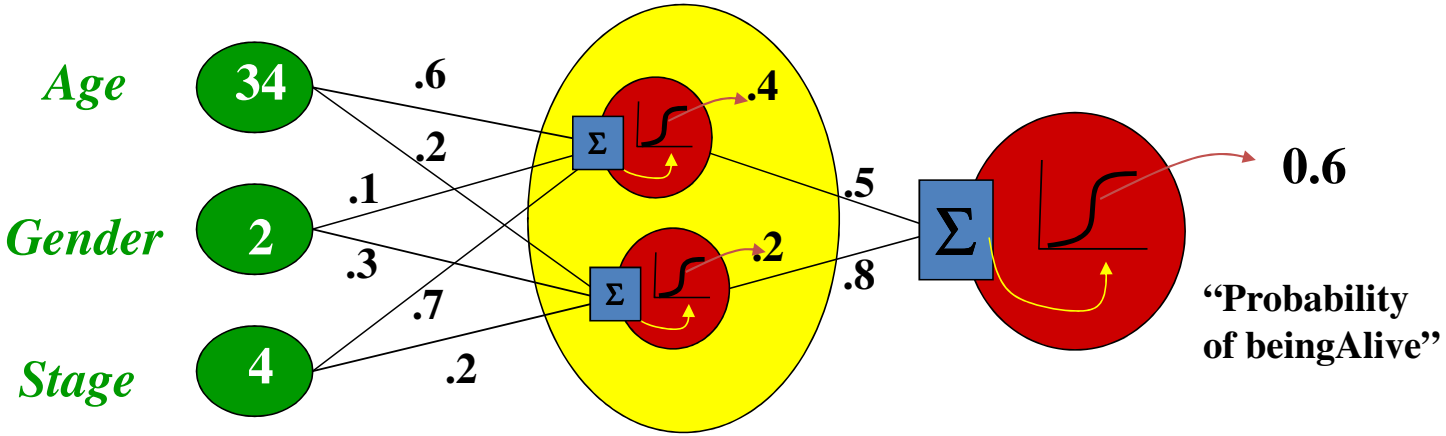


“Combined logistic models”



Overall structure

Turn weights and thresholds such that the neural networks will give desired prediction results given input.



Learning parameters in logistic regression

- Find θ, such that the conditional likelihood of the labels is maximized

$$\max_{\theta} l(w) := \log \prod_{i=1}^{m} P\left(y^i \middle| x^i, w\right)$$

- Good news: $l(w)$ is concave function of w, and there is a single global optimum.

- Bad new: no closed form solution (resort to numerical method)

Training/fitting neural networks: Backpropagation

Find w, α, β, such that the value of the output unit are close to the actual labels (0 or 1)

$$\min_{w,\alpha,\beta} l(w, \alpha, \beta) := \sum_{i}^{n} \left(y^i - \sigma\left(w^\top z^i\right)\right)^2$$

where (use sigmoid function σ)

$$z_1^i = \sigma\left(\alpha^\top x^i\right)$$
$$z_2^i = \sigma(\beta^\top x^i)$$

Non-convex objective function

Use gradient decent to find a local optimum

The gradient of $l(w, \alpha, \beta)$

$l(w, \alpha, \beta) := \sum_{i=1}^{n} \left(y^i - \sigma\left(w^\top z^i\right)\right)^2$

where $z_1^i = \sigma\left(\alpha^\top x^i\right), z_2^i = \sigma(\beta^\top x^i)$

$$\frac{\partial \sigma(u)}{\partial u} = \sigma(u)\left(1 - \sigma(u)\right)$$

Let $u^i = w^\top z^i$

Gradient

$$\frac{\partial l(w, \alpha, \beta)}{\partial w} = -\sum_i 2\left(y^i - \sigma\left(u^i\right)\right)\sigma\left(u^i\right)\left(1 - \sigma\left(u^i\right)\right) z^i$$

z^i is computed using α and β from previous iteration

$$z_1^i = \sigma\left(\alpha^\top x^i\right), z_2^i = \sigma(\beta^\top x^i)$$

The gradient with respect to α, β

Use chain rule of derivatives (backpropagation)

The "error" at layer k depends on the error at the next layer k+1

Note $z_1^i = \sigma(\alpha^\top x^i), z_2^i = \sigma(\beta^\top x^i)$

Let $v^i = \alpha^\top x^i$

$$\frac{\partial \sigma(u)}{\partial u} = \sigma(u)\big(1 - \sigma(u)\big)$$

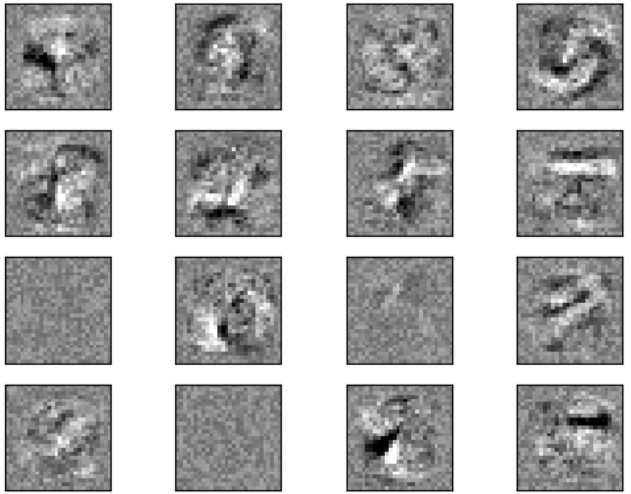
Gradient

$$\frac{\partial l(w, \alpha, \beta)}{\partial \alpha} = \frac{\partial l(w, \alpha, \beta)}{\partial z_1^i} \frac{\partial z_1^i}{\partial \alpha}$$

$$= -\sum_i 2\left(y^i - \sigma(u^i)\right)\sigma(u^i)\left(1 - \sigma(u^i)\right) w_1 \sigma(v^i)\left(1 - \sigma(v^i)\right) x^i$$

Demo: hand-written digits

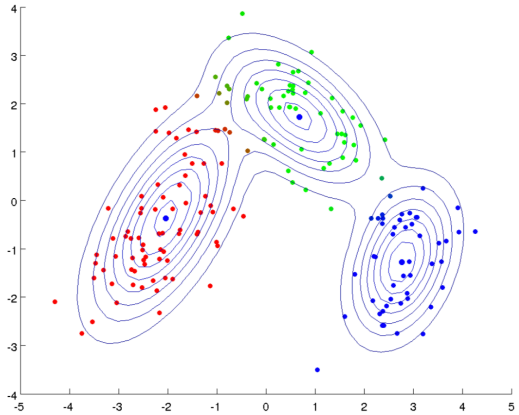
- The input data consists of 28x28 pixel handwritten digits, leading to 784 features in the dataset.
- One-hidden layer
- Therefore the first layer weight matrix have the shape (784, hidden_layer_size)
- We can therefore visualize weight for each hidden node as a 28x28 pixel image.



https://scikit-learn.org/stable/auto_examples/neural_networks/plot_mnist_filters.html#sphx-glr-auto-examples-neural-networks-plot-mnist-filters-py

Demo: Classification of wine data

- Data: chemical analysis of wines grown in the same region in Italy but derived from three different cultivars. 13 features, three types of wines.
- Use multi-class classification in the last layer



Classification performance measures

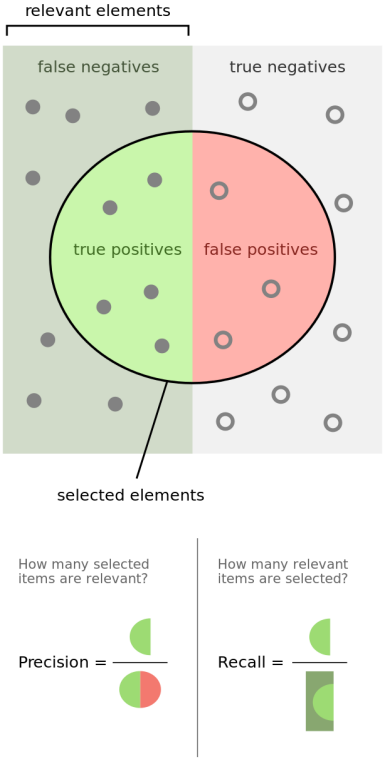
- Mis-classification error:

  # mis-classified samples / # total

- Confusion matrix

a sample of 13 pictures, 8 of cats and 5 of dogs

		Actual class	
		Cat	**Dog**
Predicted class	**Cat**	5	2
	Dog	3	3



Source: wikipedia

Classification performance measures

- Precision and recall

Precision =

retrieved relevant instances / # total retrieved

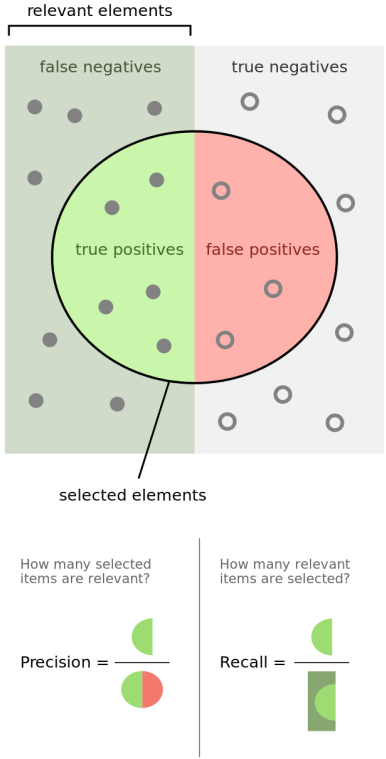
Recall =

retrieved relevant instances / # total relevant

F1 score = $\frac{2}{\text{precision}^{-1}+\text{recall}^{-1}} \in [0,1]$

- Using the previous example

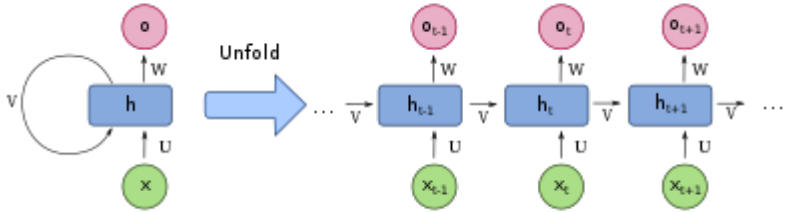
	Precision	Recall	F1-score
Cat	0.714	0.625	0.666
Dog	0.500	0.600	0.545



Source: wikipedia

Some common neural network structures

- Full connected (all nodes in the k-th layer are connected to all nodes in the (k+1)-th layer (not practical)
- Convolutional neural networks (nodes in one layer are only “locally” connected to nodes in the next layer), two types of layers: convolution and max pooling
- Recursive neural network (RNN): good for sequential / time series data; neural network structure “repeats” itself; attention mechanism, transformer

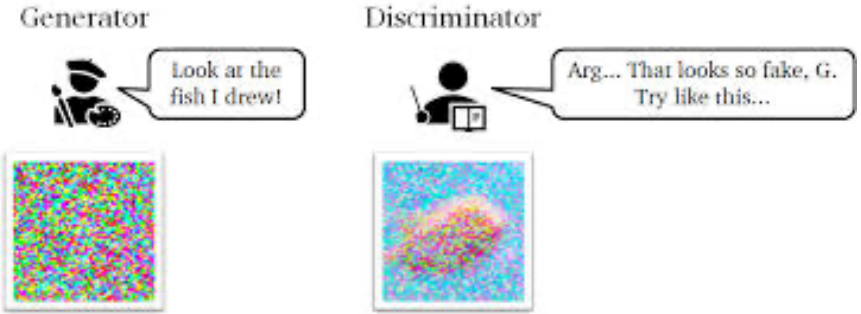


- Residual neural networks: jump over some layers

GAN

- Generative adversarial neural networks (GAN) (2014) – generative model and better discriminator

$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))].$$

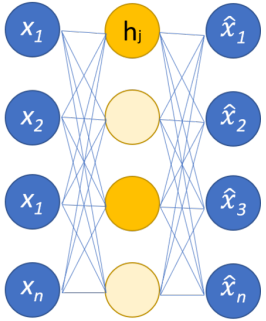


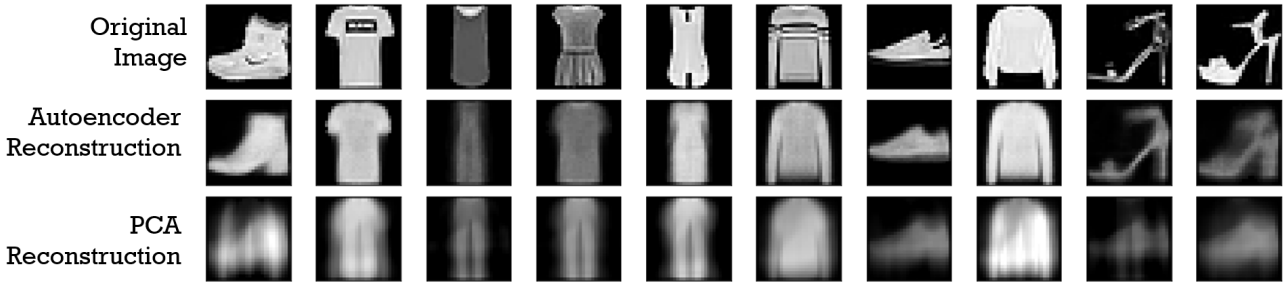
An image generated by a StyleGAN that looks deceptively like a photograph of a real person. This image was generated by a StyleGAN based on an analysis of portraits.

https://towardsdatascience.com/gan-papers-to-read-in-2020-2c708af5c0a4

Auto-encoder

- Suppose we only have unlabeled training samples
- Autoencoder is a neural networks for unsupervised learning: setting the target value to input
- Network learn to represent data itself
- Variational autoencoder (VAE) gives generative model
- If linear activations are used, or only a single sigmoid hidden layer, then the optimal solution to an autoencoder is strongly related to PCA

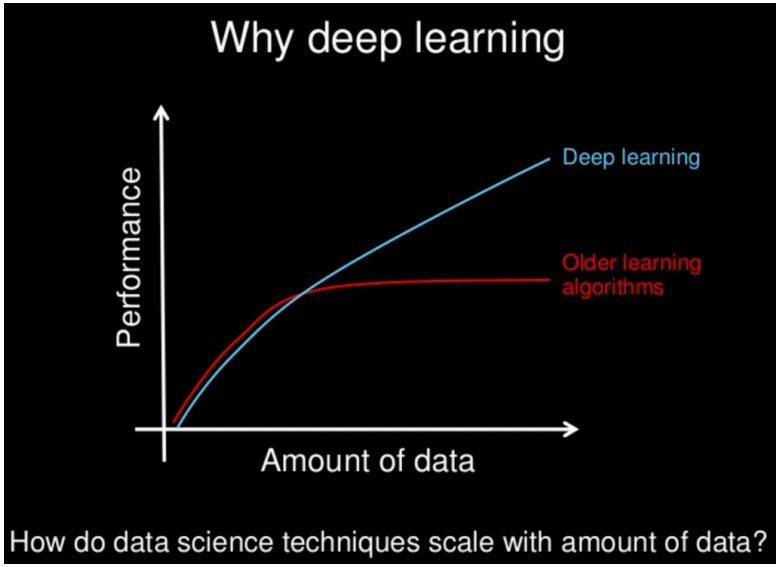


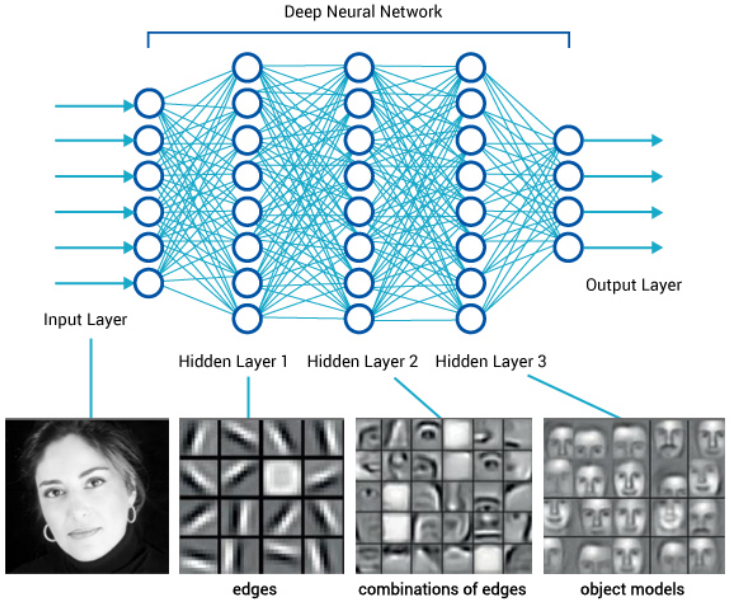


Reconstruction of 28x28pixel images by an Autoencoder with a code size of two (two-units hidden layer) and the reconstruction from the first two Principal Components of PCA. Images come from the Fashion MNIST dataset.[31]

Why deep learning

- Deep learning (deep neural networks) revolutionary

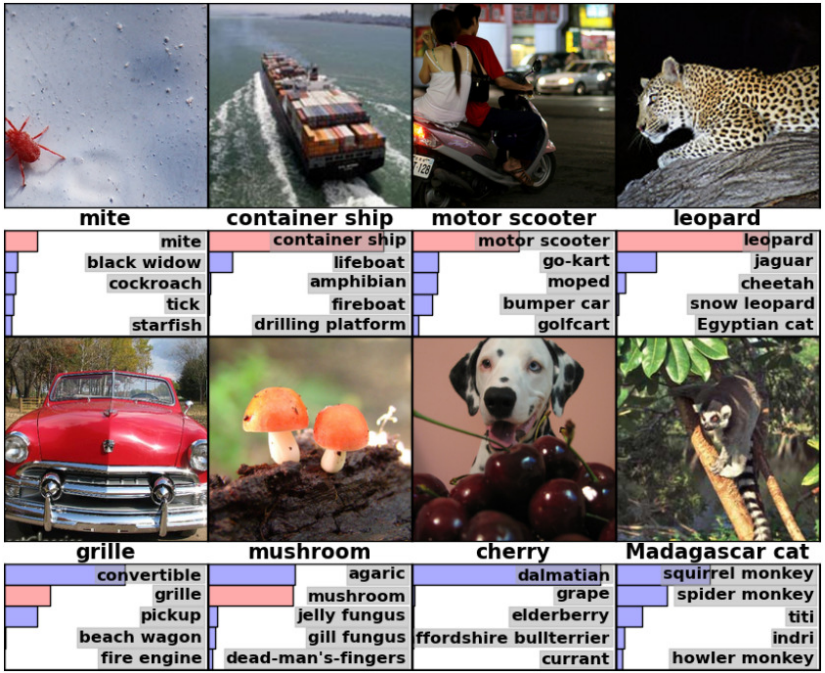




ImageNet

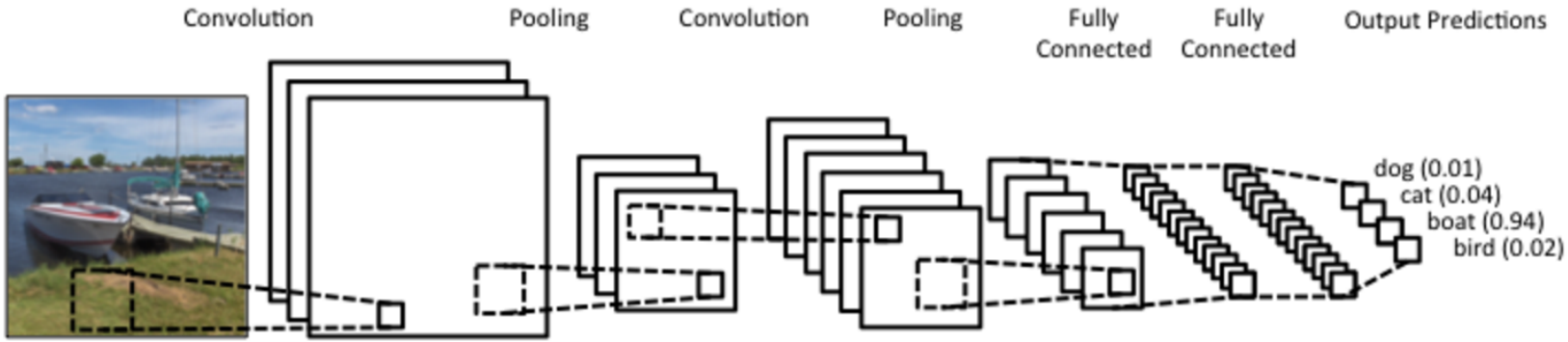
Image classification with 1.3M color images and 1000 classes

Need large scale nonparametric methods



https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

Convolutional Neural Network (CNN)



https://www.cs.ryerson.ca/~aharley/vis/conv/

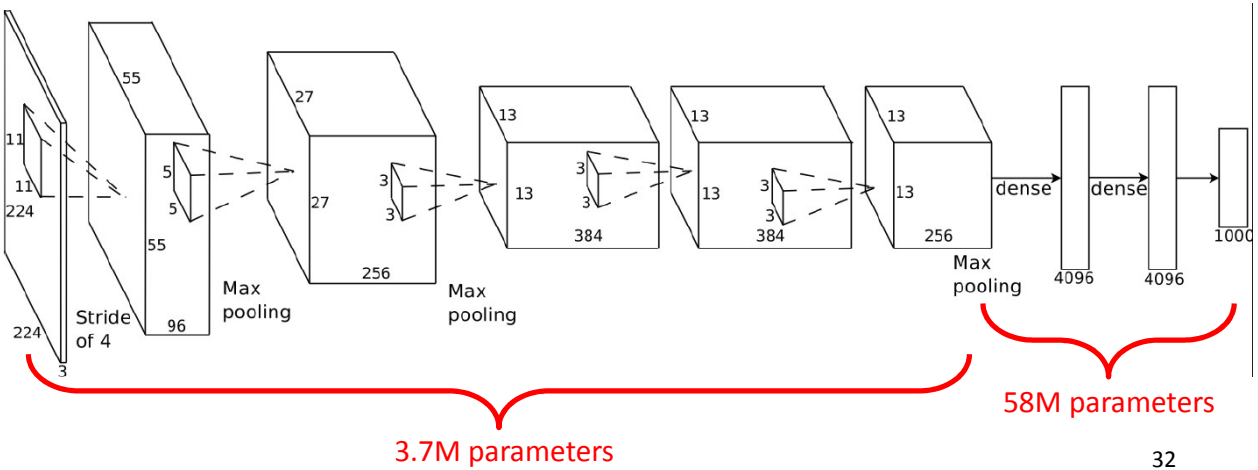
Convolutional Neural Network (CNN)

8 layer convolution neural network [Krizhevsky12], Achieved state-of-the-art result (beating the second place by 10%)

First 5 layers: convolution + max pooling

Next 2 layers: fully connected nonlinear neurons

Last layer: multiclass logistic regression

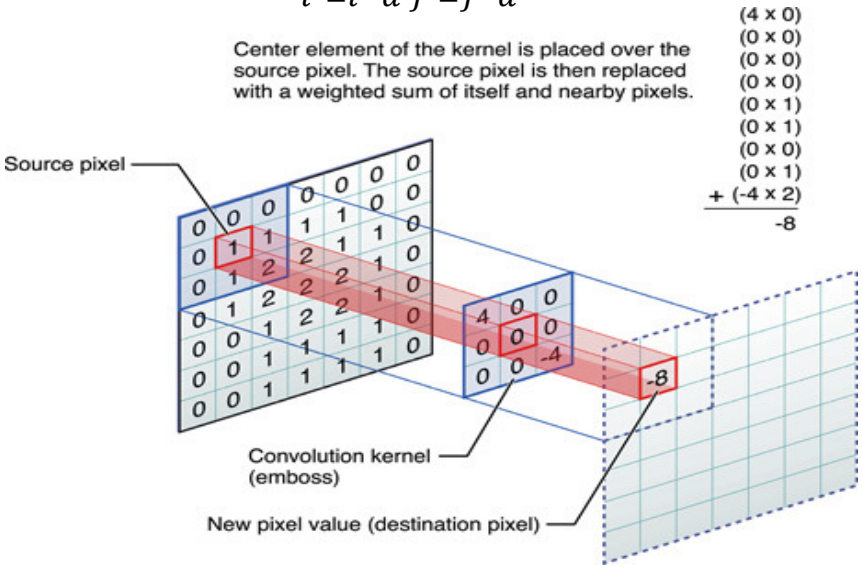


https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

Convolution layer

- Source image *I*, destination image *O*, convolving image with a kernel *K* of size *2d+1*

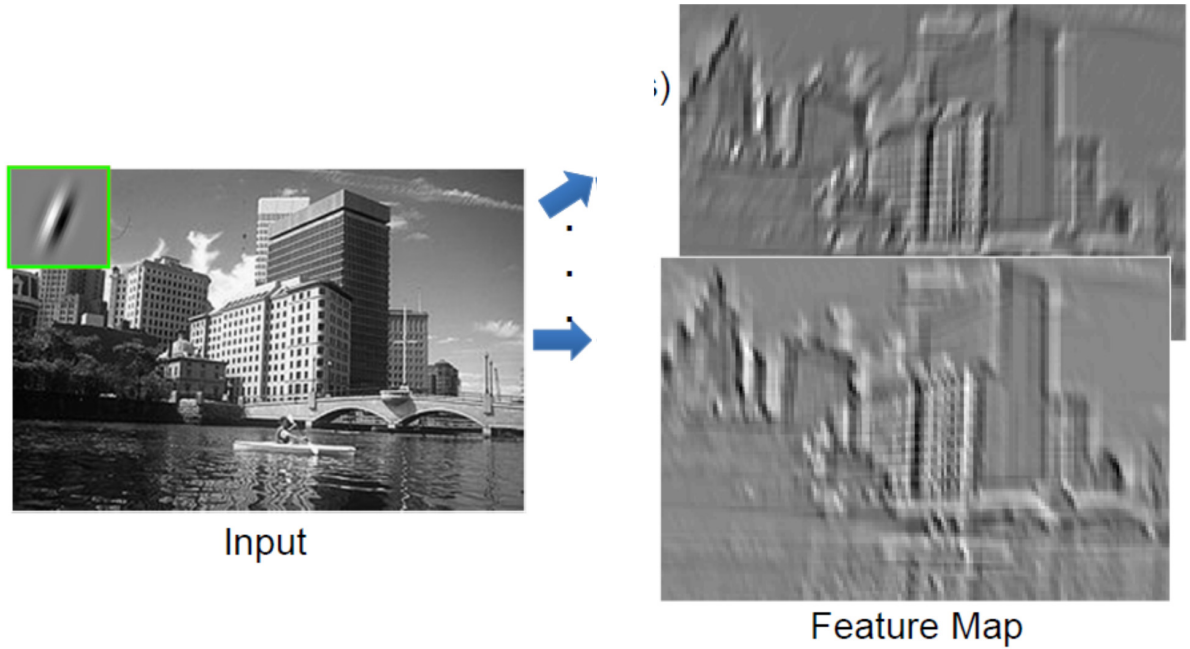
$$O(i,j) = \sum_{i'=i-d}^{i+d} \sum_{j'=j-d}^{j+d} I(i',j')K(i',j')$$



https://www.embedded-vision.com/sites/default/files/technicalarticles/ARMFFT/2014Article/kernel_convolution.jpg

Convolutional feature

Convolution extract local image features

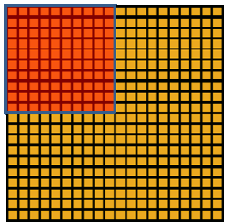


https://i.imgur.com/TqTJCrz.png

Convolutional Neural Network

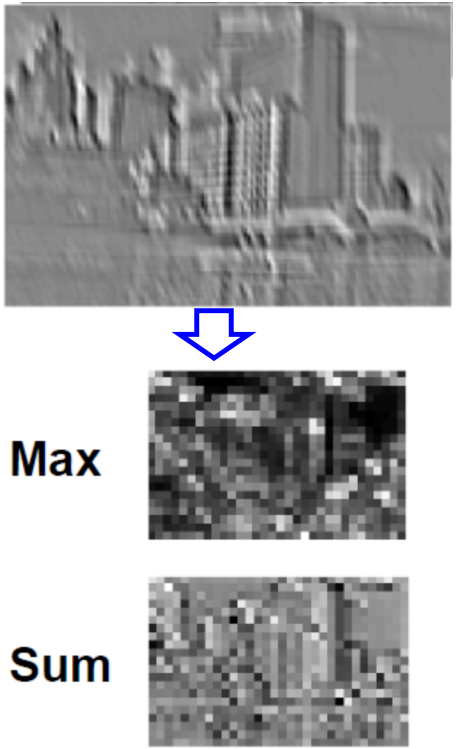
Spatial pooling

- Sum or max (usually max)
- Overlapping or non-overlapping regions
- Invariant to small translations



Convolved feature

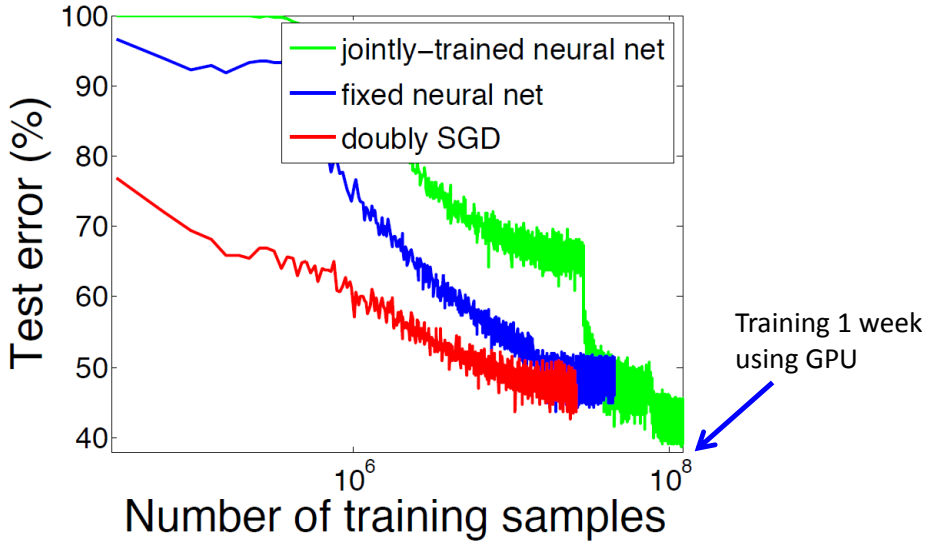
Pooled feature



https://yoyoinwanderland.github.io/Deep-Learning-1/

ImageNet

Millions of training points, 1000 class classification task



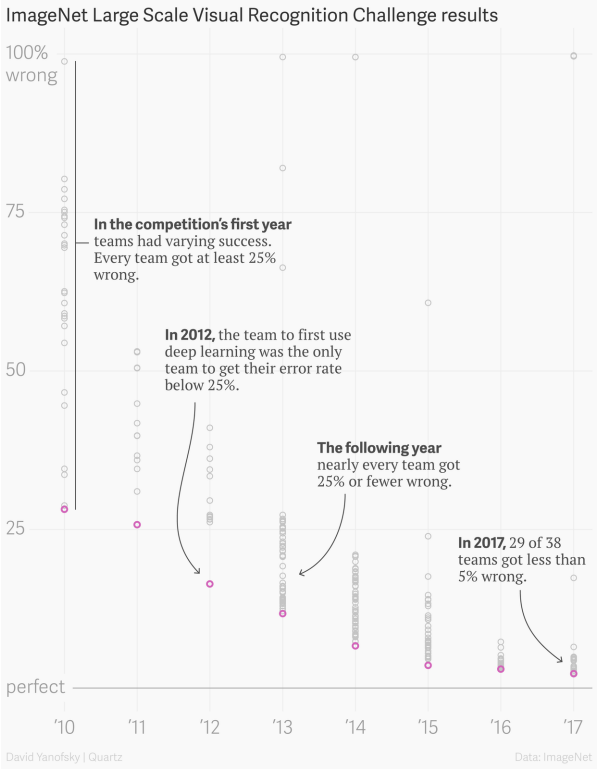
https://github.com/zixu1986/Doubly_Stochastic_Gradients

Georgia Tech

ImageNet: Demonstrate power of deep learning

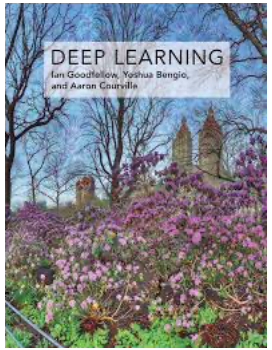
On 30 September 2012, a convolutional neural network (CNN) called AlexNet achieved a top-5 error of **15.3%** in the ImageNet 2012 Challenge, more than 10.8 percentage points lower than that of the runner up. This was made feasible due to the use of Graphics processing units (GPUs) during training, an essential ingredient of the deep learning revolution.

According to The Economist, "Suddenly people started to pay attention, not just within the AI community but across the technology industry as a whole."



http://www.image-net.org

Ending words



2015

- Deep learning is the “hot-topic” in “hot-topics”
- Rapidly evolving
- Many research and engineering efforts, hugely successful in various domain: image, speech etc.
- Many remaining questions: interpretability (black-box), reliability (e.g., healthcare, manufacturing)
- Additional references
- NeurIPS 2015 tutorial on deep learning: https://aiatadams.files.wordpress.com/2016/02/bengio-lecun-20151207-deep-learning-tutorial-nips.pdf
- NeurIPS 2016 tutorial on GAN: https://media.nips.cc/Conferences/2016/Slides/6202-Slides.pdf
- Deep learning book: https://www.deeplearningbook.org


SUMMARY