Computational Data Analysis

Machine Learning

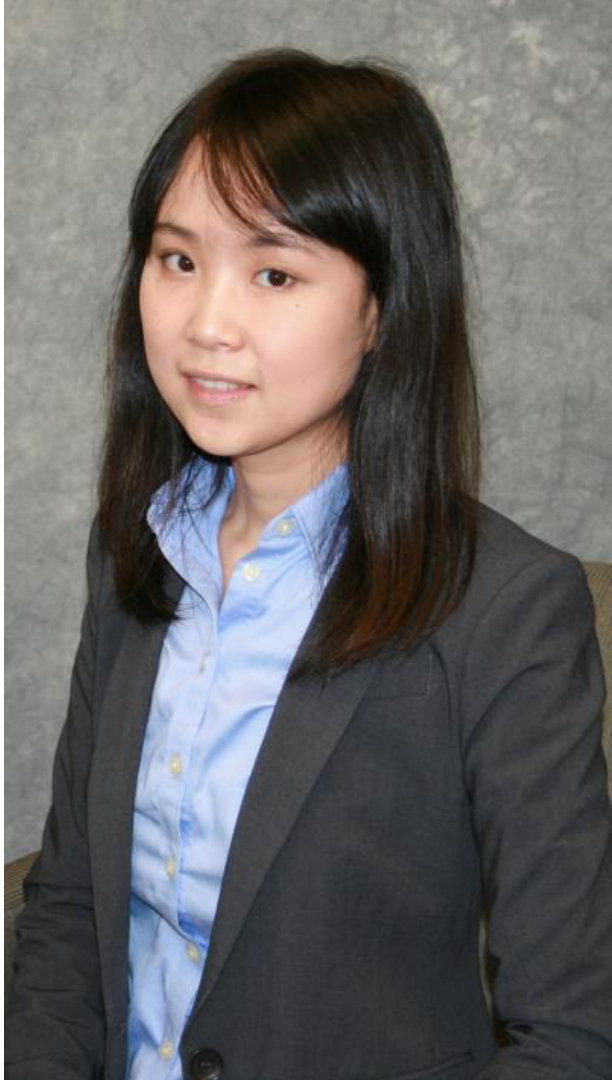
Yao Xie, Ph.D.

Associate Professor

Harold R. and Mary Anne Nash Early Career Professor

H. Milton Stewart School of Industrial and Systems Engineering

Introduction to Neural Networks

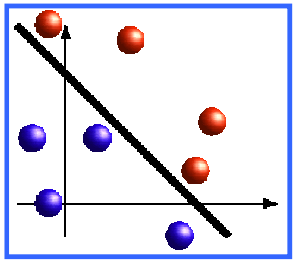


Main approaches to design classifiers

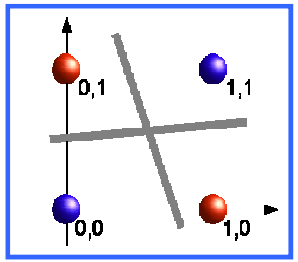
- Bayes rule, use simplifying assumption for $p(x|y=1)$
  - Assume $p(x|y=1)$ is Gaussian
  - Assume $p(x|y=1)$ is fully factorized
- Use geometric intuitions
  - k-nearest neighbor classifier
  - Support vector machine
- Directly go for the decision boundary $\mathrm{h(x)} = -\ln \frac{q_i(x)}{q_j(x)}$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

Learning nonlinear decision boundary

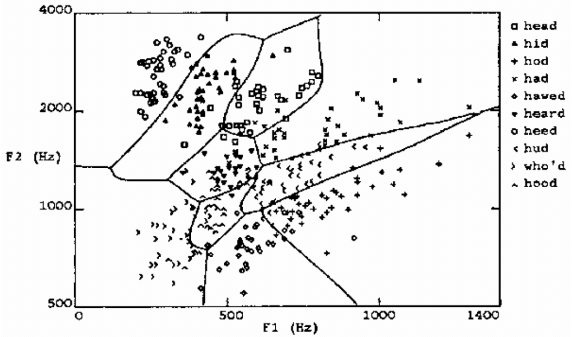
- Linearly separable



- Nonlinearly separable



The XOR gate

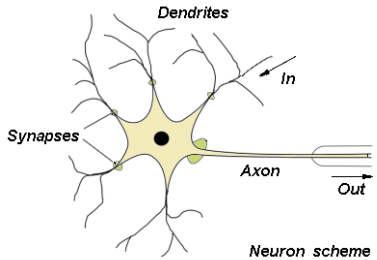


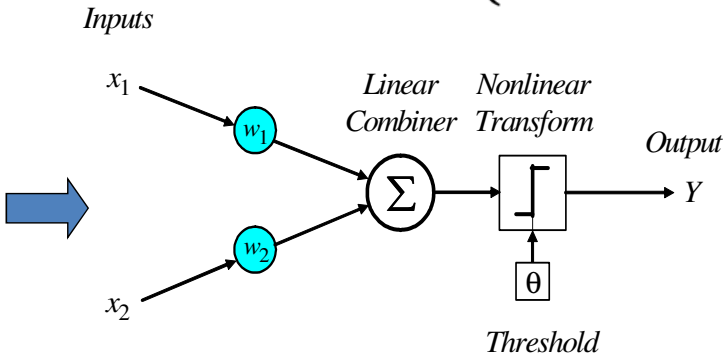
Speech recognition

Perceptron

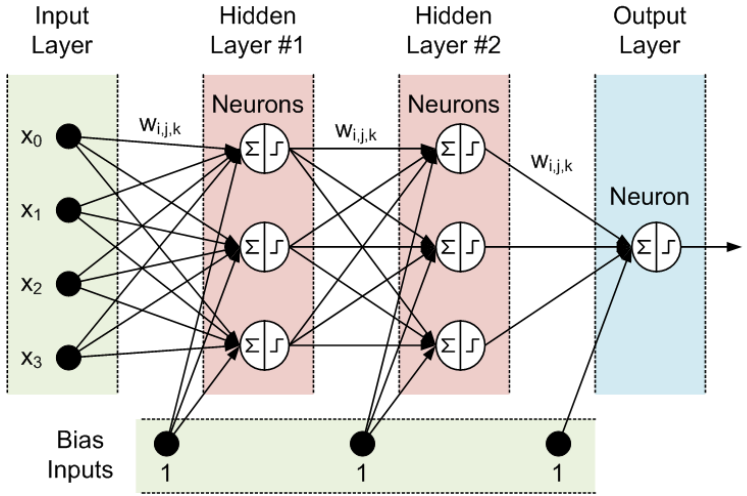
$$f(\mathbf{x}) = \begin{cases} 1 & \text{if } \mathbf{w} \cdot \mathbf{x} + b > 0, \\ 0 & \text{otherwise} \end{cases}$$

- From biological neuron to artificial neuron (perceptron)





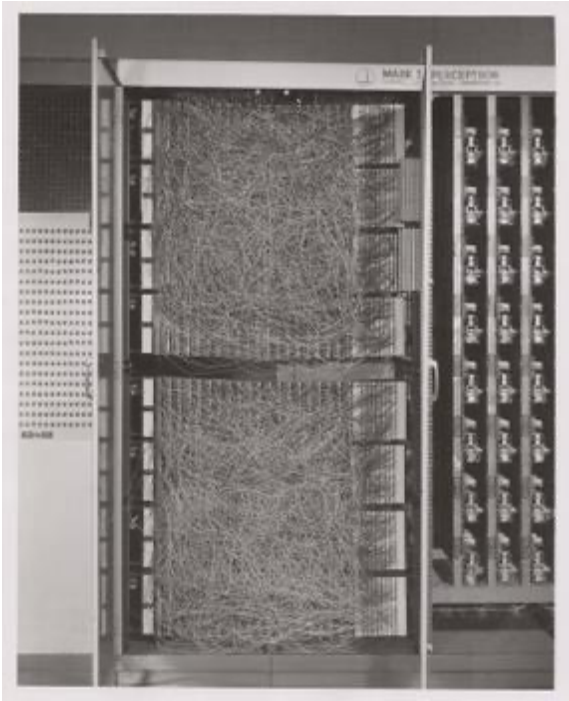
- Artificial neural nets (ANN)
  - Many neuron-like threshold switching units
  - Many weighted interconnections among units
  - Highly parallel, distributed processes



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".

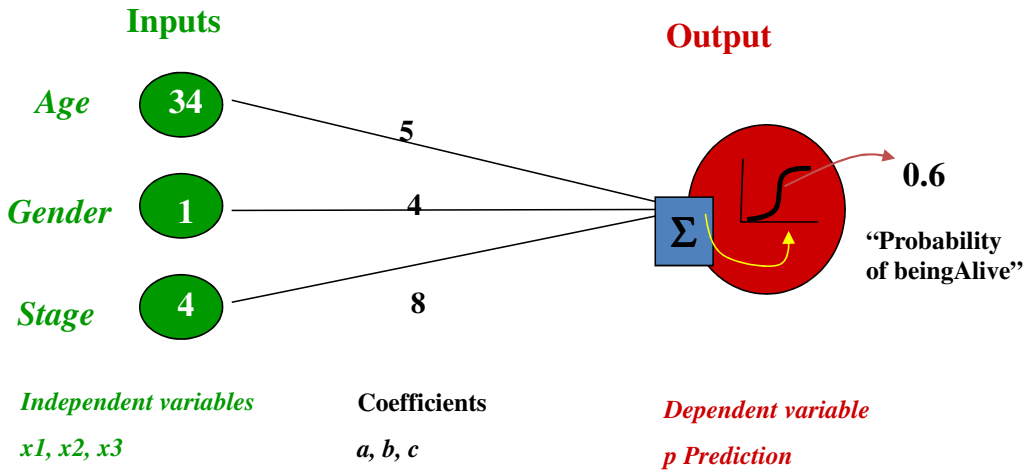


Mark I Perceptron machine, the first implementation of the perceptron algorithm. It was connected to a camera with 20×20 cadmium sulfide photocells to make a 400-pixel image. The main visible feature is a patch panel that set different combinations of input features. To the right, arrays of potentiometers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

Logistic regression can be viewed as probabilistic perceptron

- Independent variable = input variable
- Dependent variable = output variable
- Coefficients = “weights”

Logistic Regression Model (the sigmoid unit) is a perceptron

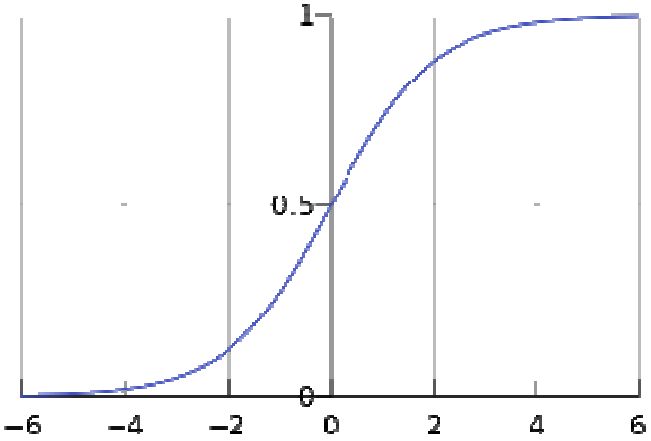


Logistic regression

- Assume that the posterior distribution $p(y = 1|x)$ take a particular form

$$p(y = 1|x, w) = \frac{1}{1 + \exp(-w^{\mathsf{T}} x)}$$

- Logistic function (or sigmoid function) $\sigma(u) = \frac{1}{1+\exp(-u)}$
  - $\frac{\partial \sigma(u)}{\partial u} = \sigma(u)\big(1 - \sigma(u)\big)$



Learning parameters in logistic regression

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

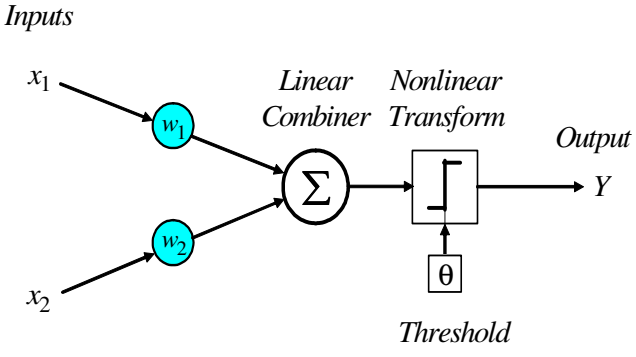
$$\max_{\theta} l(w) := \log \prod_{i=1}^{m} P\left(y^i \middle| x^i, w\right)$$

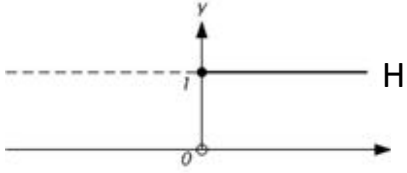
- Good news: $l(w)$ is concave function of w, and there is a single global optimum.

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

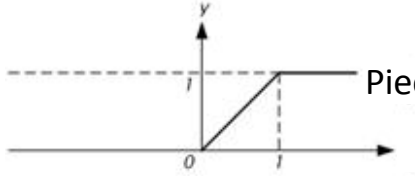
Other nonlinear neurons

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

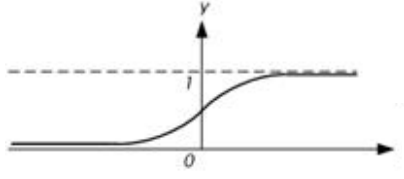




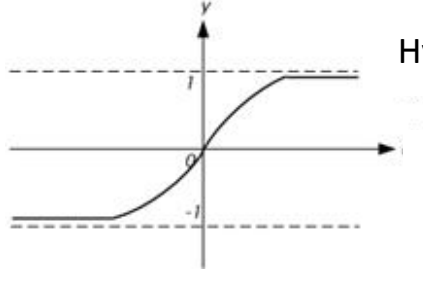
Hard limiter



Piecewise linear



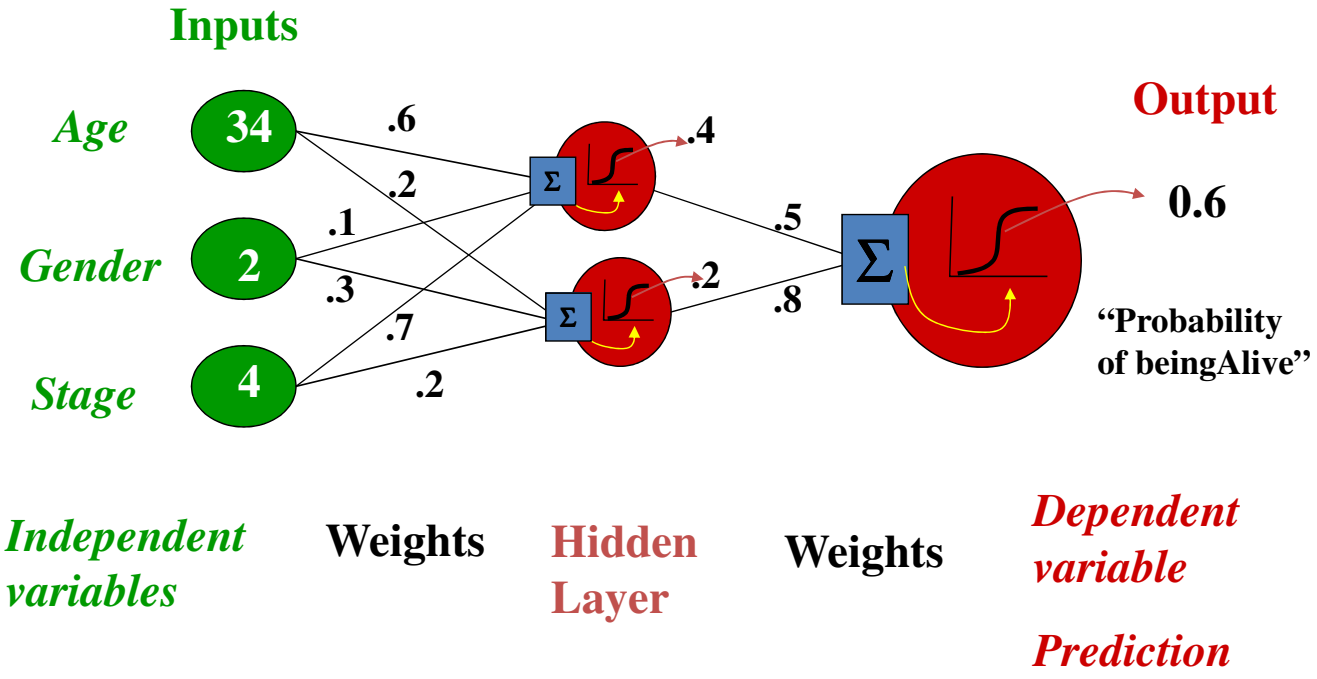
Sigmoid



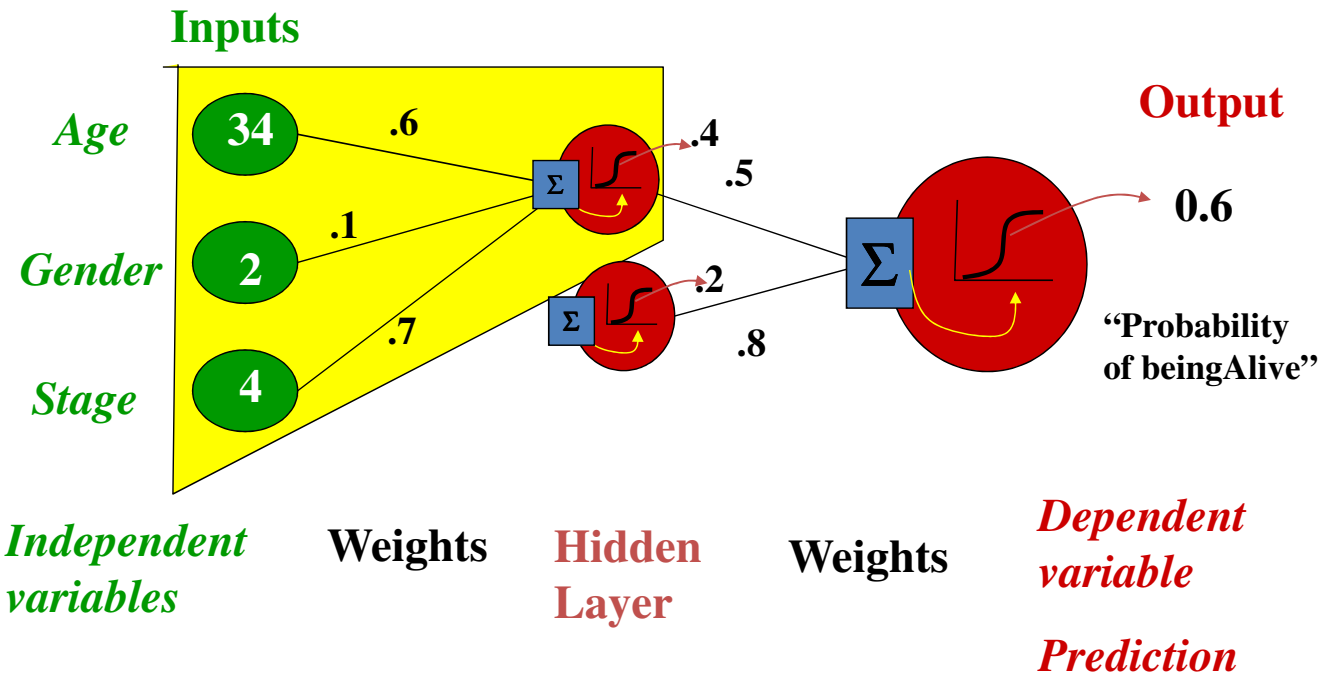
Hyperbolic tangent

Deep Neural Network Model

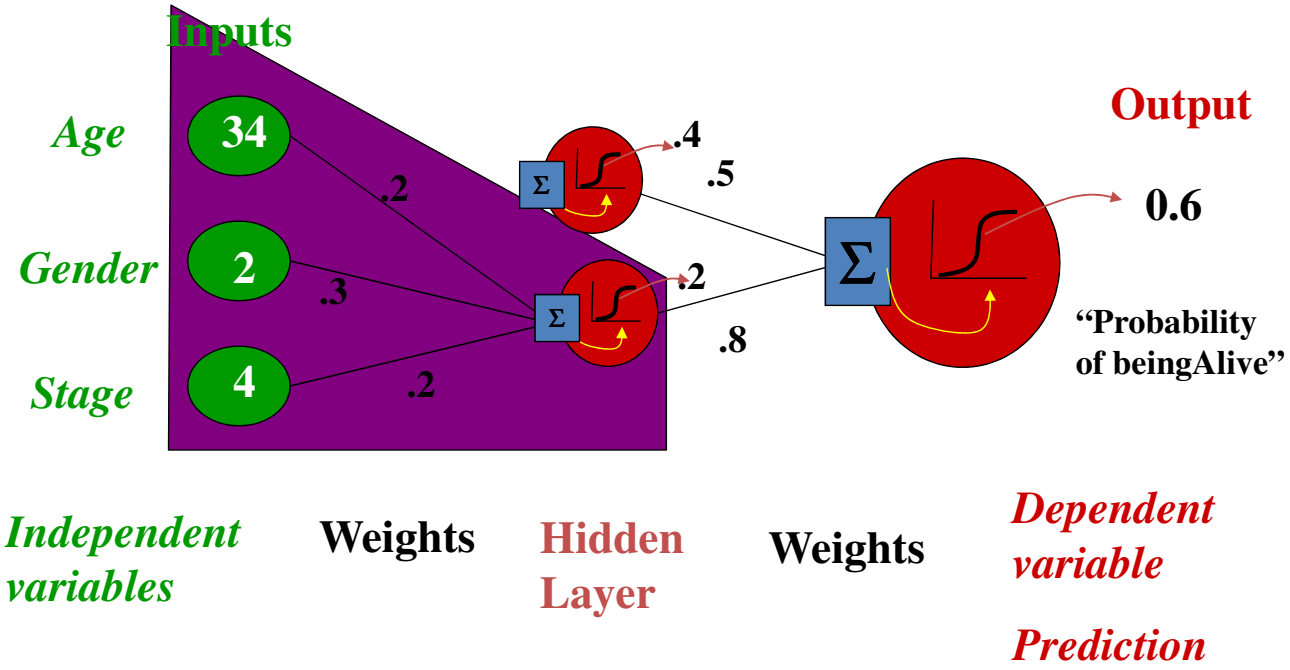
Multiple layers of interconnected perceptrons



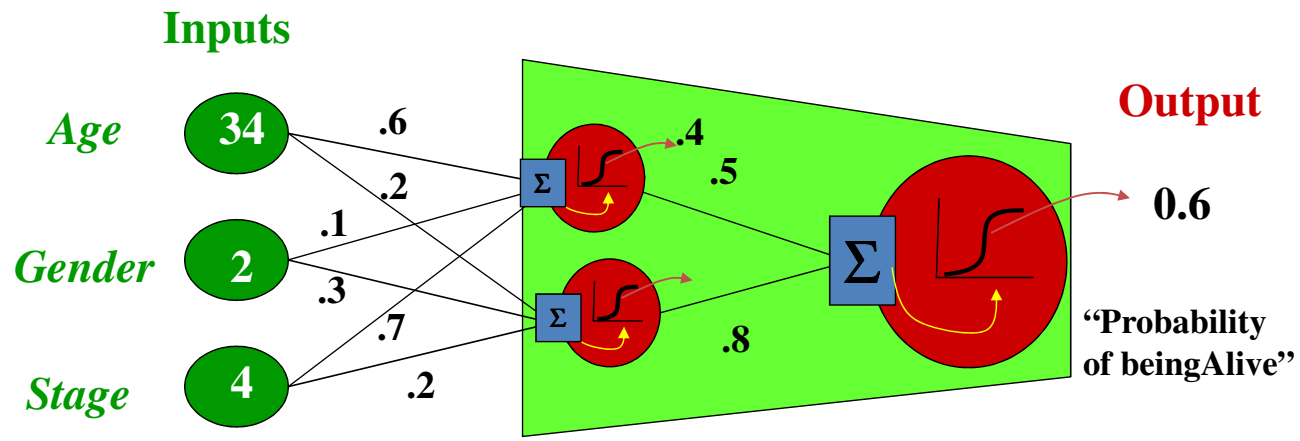
“Combined logistic models”



“Combined logistic models”

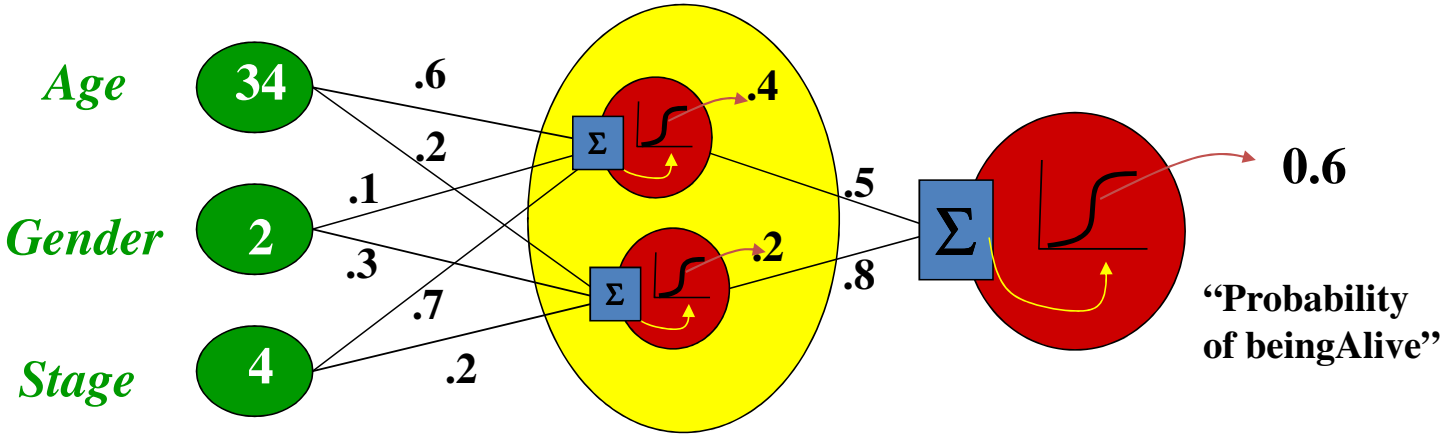


“Combined logistic models”



Overall structure

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



Expressive Capabilities of Neural Networks

- 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].

Training/fitting neural networks: Backpropagation

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

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

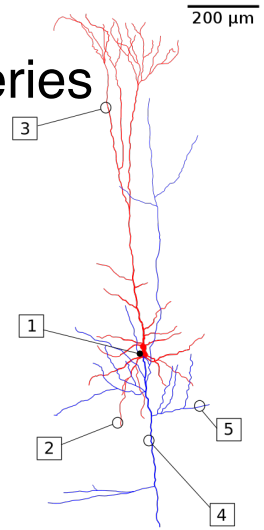
Gradient

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

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

Some common neural network structures

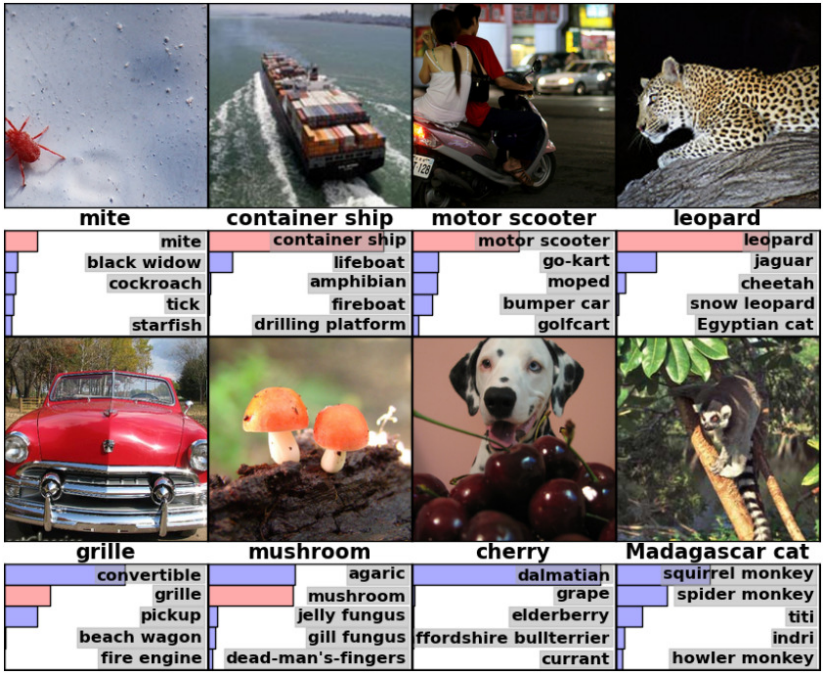
- Full connected (all nodes in the k-th layer are connected to all nodes in the (k+1)-th layer (not practical)
- Convolutional neural networks (nodes in one layer are only ”locally” connected to nodes in the next layer)
- Recursive neural network (RNN): good for sequential / time series data; neural network structure ”repeats” itself.
- Residual neural networks
- Generative adversarial neural networks (GAN)



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

Georgia Tech

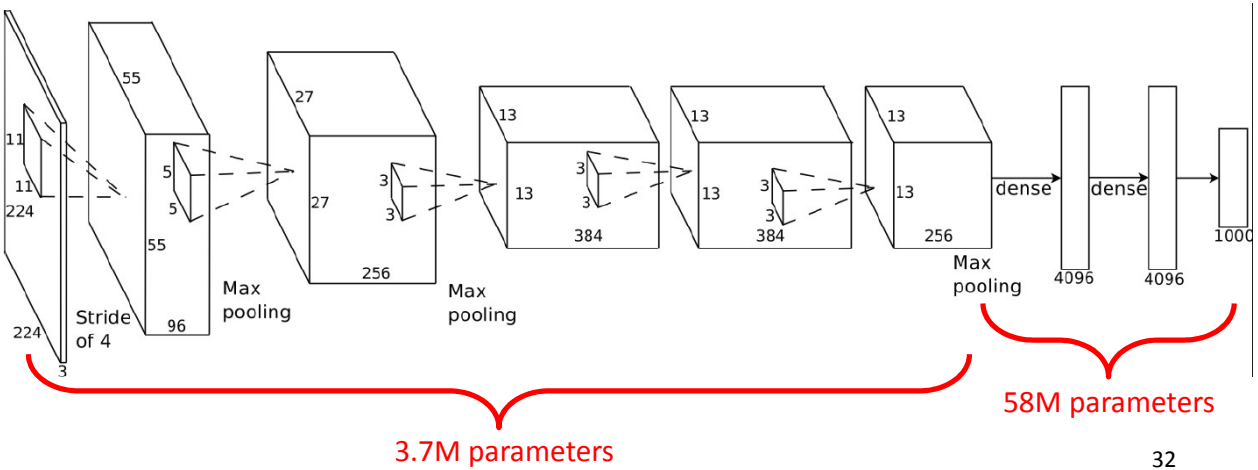
Convolutional Neural Network (CNN)

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

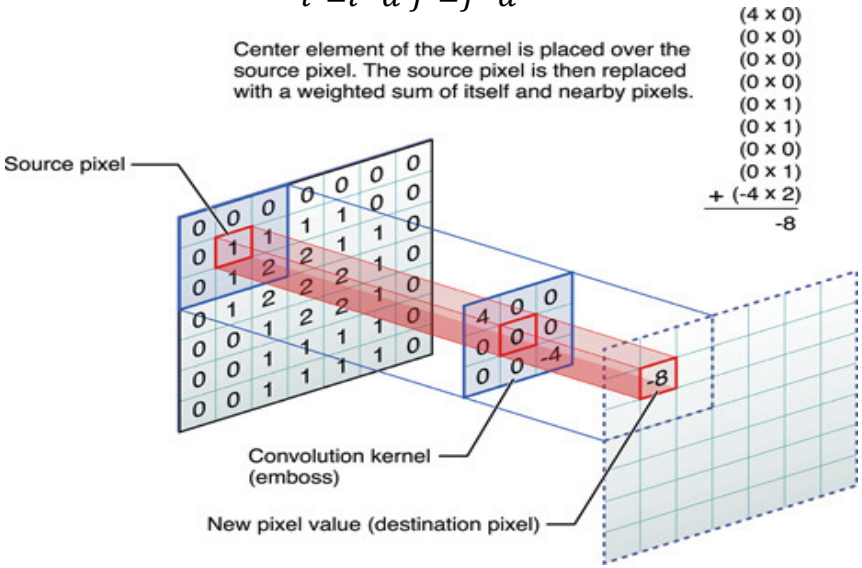


https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

Convolution layer

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

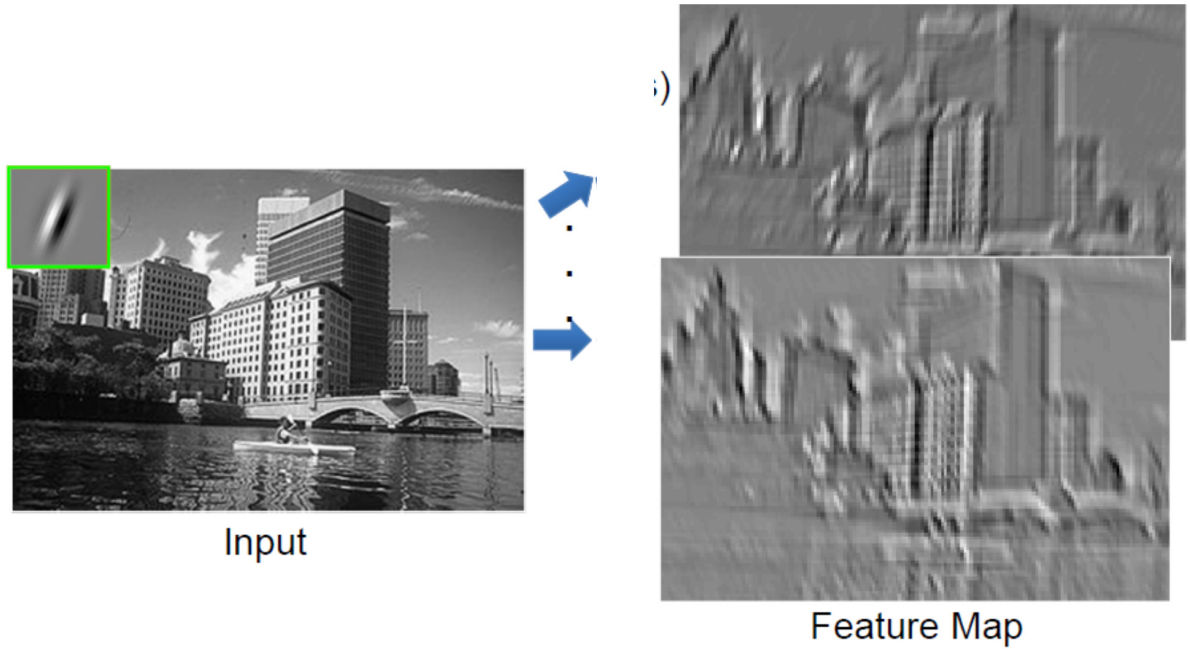
$$O(i,j) = \sum_{i'=i-d}^{i+d} \sum_{j'=j-d}^{j+d} I(i',j')K(i',j')$$



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

Convolutional feature

Convolution extract local image features

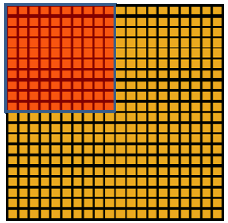


https://i.imgur.com/TqTJCrz.png

Convolutional Neural Network

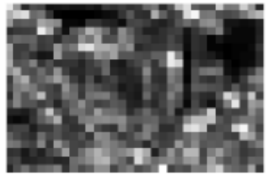
Spatial pooling

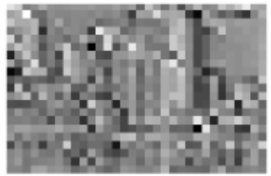
- Sum or max (usually max)
- Overlapping or non-overlapping regions
- Invariant to small translations



Convolved feature

Pooled feature



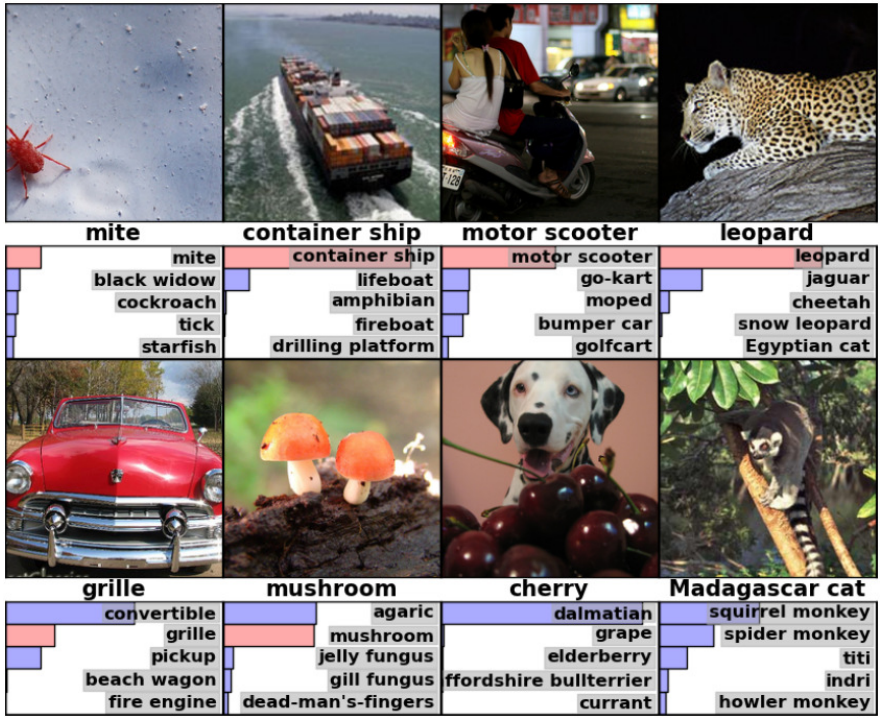


https://yoyoinwanderland.github.io/Deep-Learning-1/

ImageNet

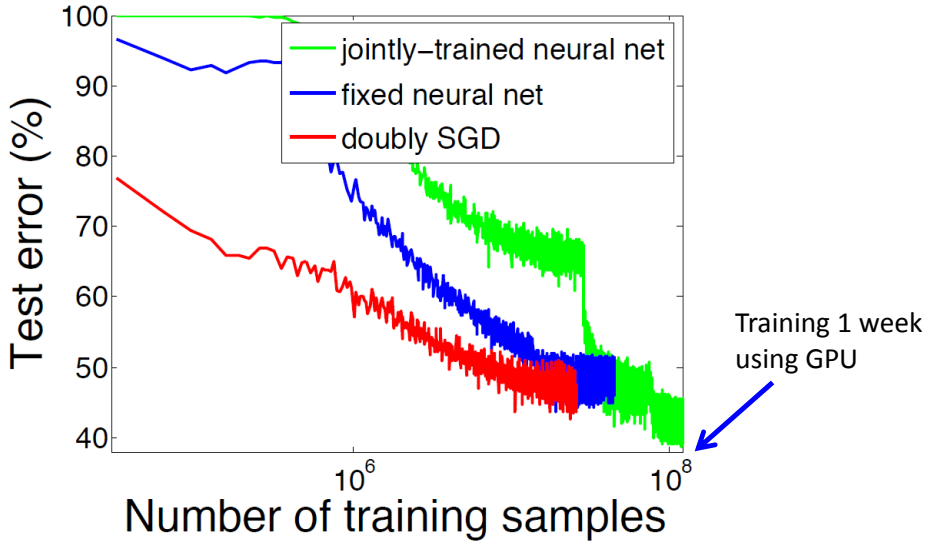
Image classification with 1.3M color images and 1000 classes

Need large scale nonparametric methods



ImageNet

Millions of training points, 1000 class classification task



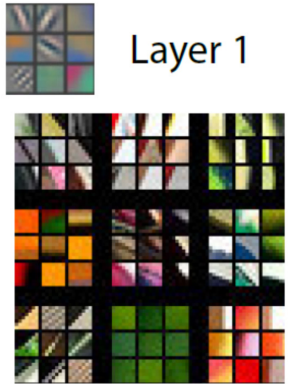
https://github.com/zixu1986/Doubly_Stochastic_Gradients

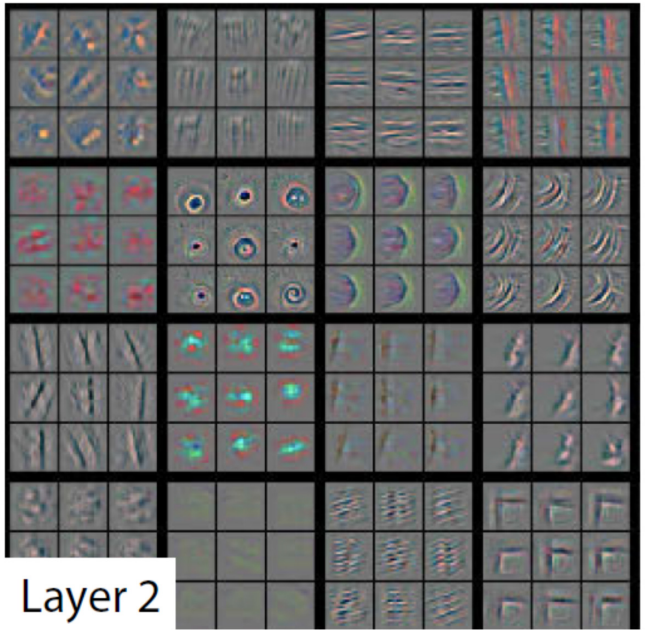
Georgia Tech

Convolutional Neural Networks

Visualize learned filters

- Lower layer learns more primitive features
- Higher layer learns more complex features





https://arxiv.org/pdf/1311.2901v3.pdf

Convolutional Neural Networks

TABLE 11.1. *Test set performance of five different neural networks on a handwritten digit classification example (Le Cun, 1989).*

	Network Architecture	Links	Weights	% Correct
Net-1:	Single layer network	2570	2570	80.0%
Net-2:	Two layer network	3214	3214	87.0%
Net-3:	Locally connected	1226	1226	88.5%
Net-4:	Constrained network 1	2266	1132	94.0%
Net-5:	Constrained network 2	5194	1060	98.4%

Ending words

- Deep learning is the ”hot-topic” in “hot-topics”
- Rapidly evolving
- Dedicated course on neural networks
- Many engineering efforts, hugely successful in various domain: image, speech etc.
- Still fairly nascent in other more complex domains: healthcare, logistics, finance.
- 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


SUMMARY