COGS500/CMPE489: Cognitive Science

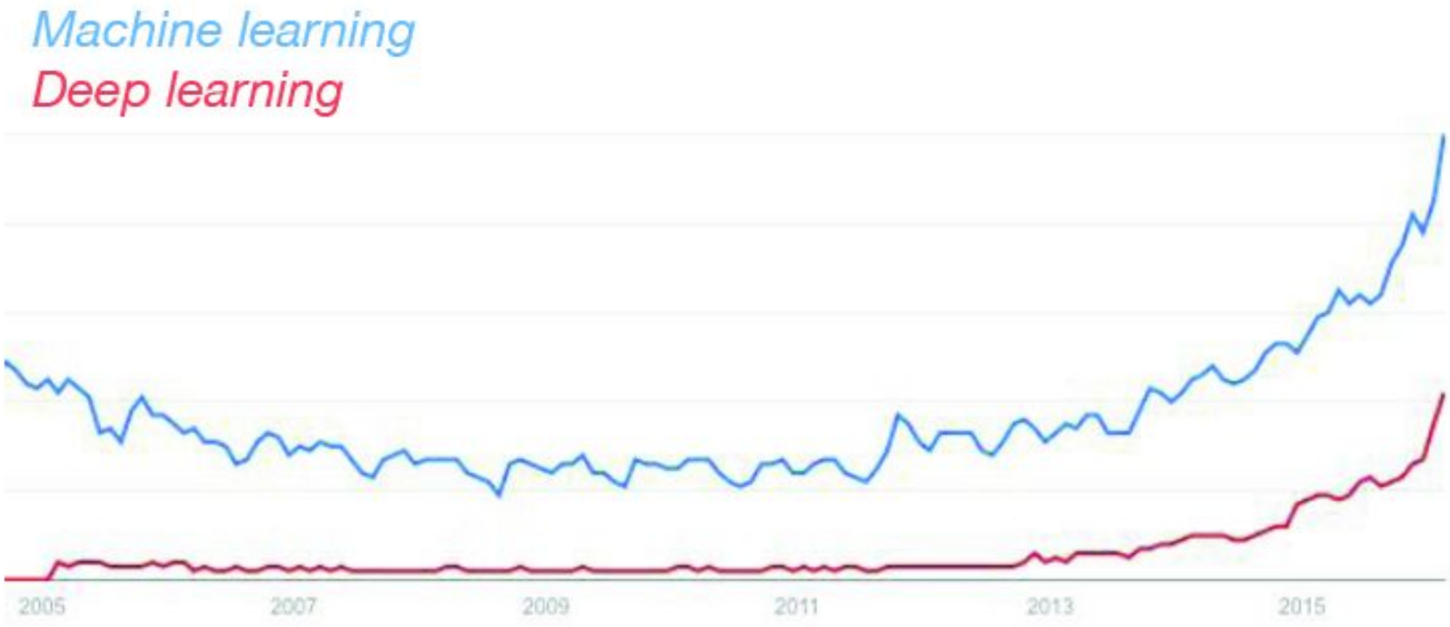
Learning

MACHINE LEARNING

Why "Learn" ?

- Machine learning is programming computers to optimize a performance criterion using example data or past experience.
- There is no need to "learn" to calculate payroll
- Learning is used when:
  - Human expertise does not exist (navigating on Mars),
  - Humans are unable to explain their expertise (speech recognition)
  - Solution changes in time (routing on a computer network)
  - Solution needs to be adapted to particular cases (user biometrics)

Google Trends



What We Talk About When We Talk About"Learning"

- Learning general models from a data of particular examples
- Data is cheap and abundant (data warehouses, data marts); knowledge is expensive and scarce.
- Example in retail: Customer transactions to consumer behavior:

  People who bought "Blink" also bought "Outliers" (www.amazon.com)
- Build a model that is *a good and useful approximation* to the data.

What is Machine Learning?

- Optimize a performance criterion using example data or past experience.
- Role of Statistics: Inference from a sample
- Role of Computer science: Efficient algorithms to
  - Solve the optimization problem
  - Representing and evaluating the model for inference

Applications

- Association
- Supervised Learning
  - Classification
  - Regression
- Unsupervised Learning
- Reinforcement Learning

Learning Associations

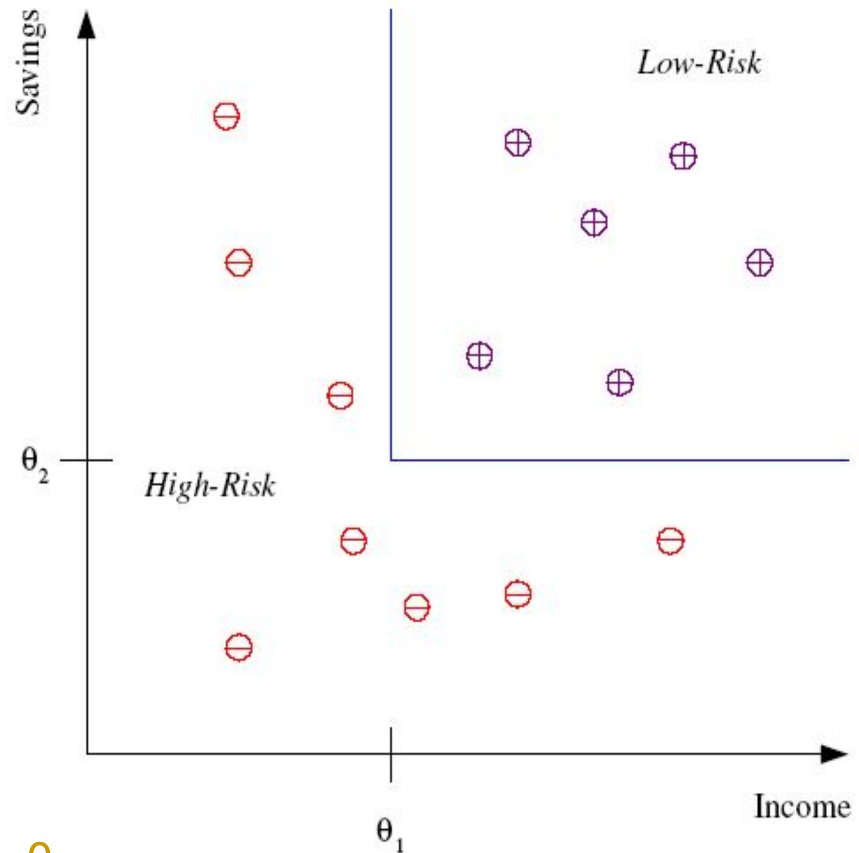
- Basket analysis:

  $P(Y \mid X)$ probability that somebody who buys X also buys Y where X and Y are products/services.

  Example: P (chips | beer) = 0.7

Classification

- Example: Credit scoring
- Differentiating between low-risk and high-risk customers from their *income* and *savings*

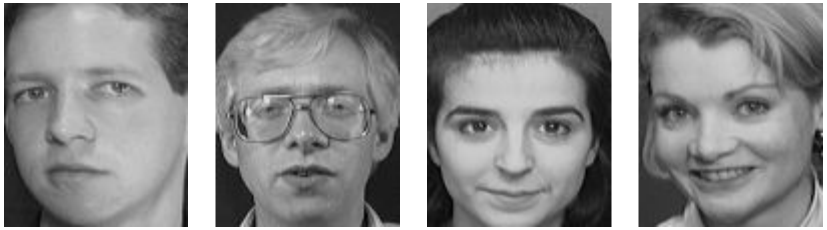


Discriminant: IF *income* > θ_1 AND *savings* > θ_2
THEN low-risk ELSE high-risk

Face recognition

Training examples of a person

Test images



ORL dataset,
AT&T Laboratories, Cambridge UK

Regression

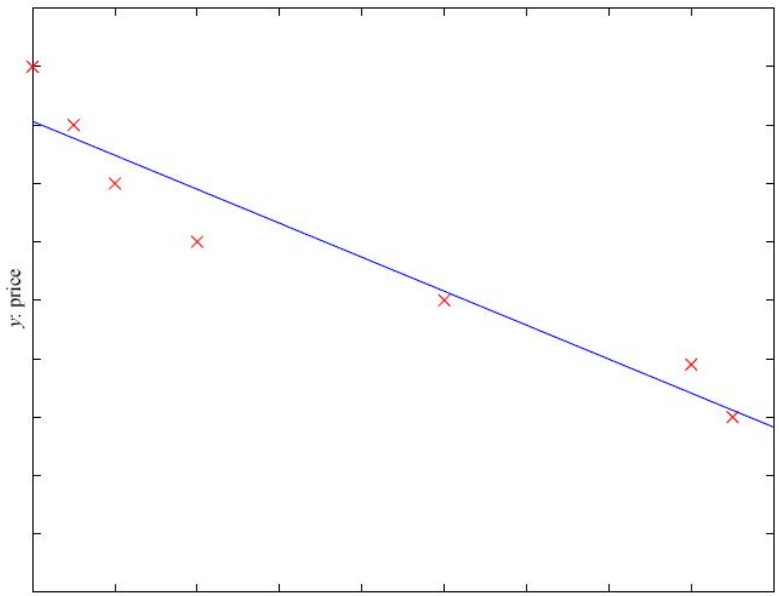
- Example: Price of a used car
- *x* : car attributes

  y : price

  $y = g(x \mid \theta)$

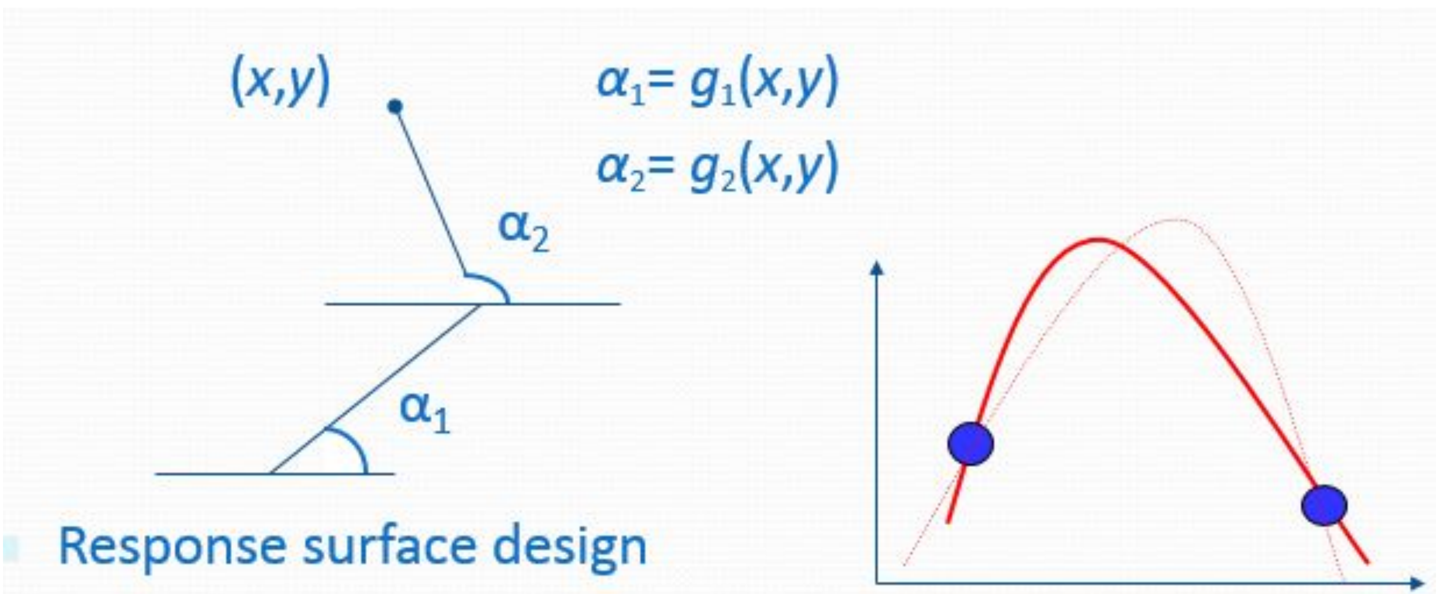
  g ( ) model,

 θ parameters



Regression Applications

- Navigating a car: Angle of the steering
- Kinematics of a robot arm



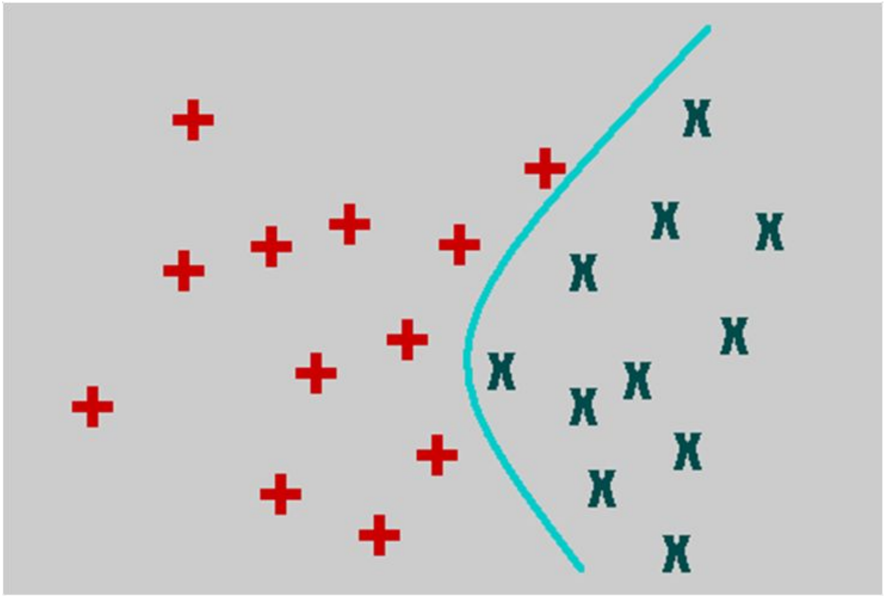
Supervised Learning: Uses

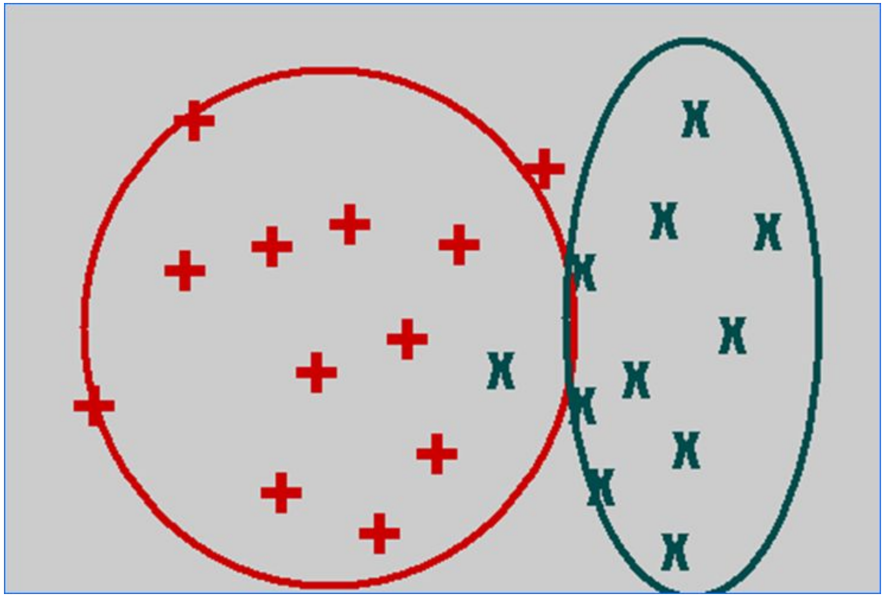
- Prediction of future cases: Use the rule to predict the output for future inputs
- Knowledge extraction: The rule is easy to understand
- Compression: The rule is simpler than the data it explains
- Outlier detection: Exceptions that are not covered by the rule, e.g., fraud

Unsupervised Learning

- Learning “what normally happens”
- No output
- Clustering: Grouping similar instances
- Example applications
  - Customer segmentation in CRM
  - Image compression: Color quantization
  - Bioinformatics: Learning motifs

Supervised vs. Unsupervised Learning

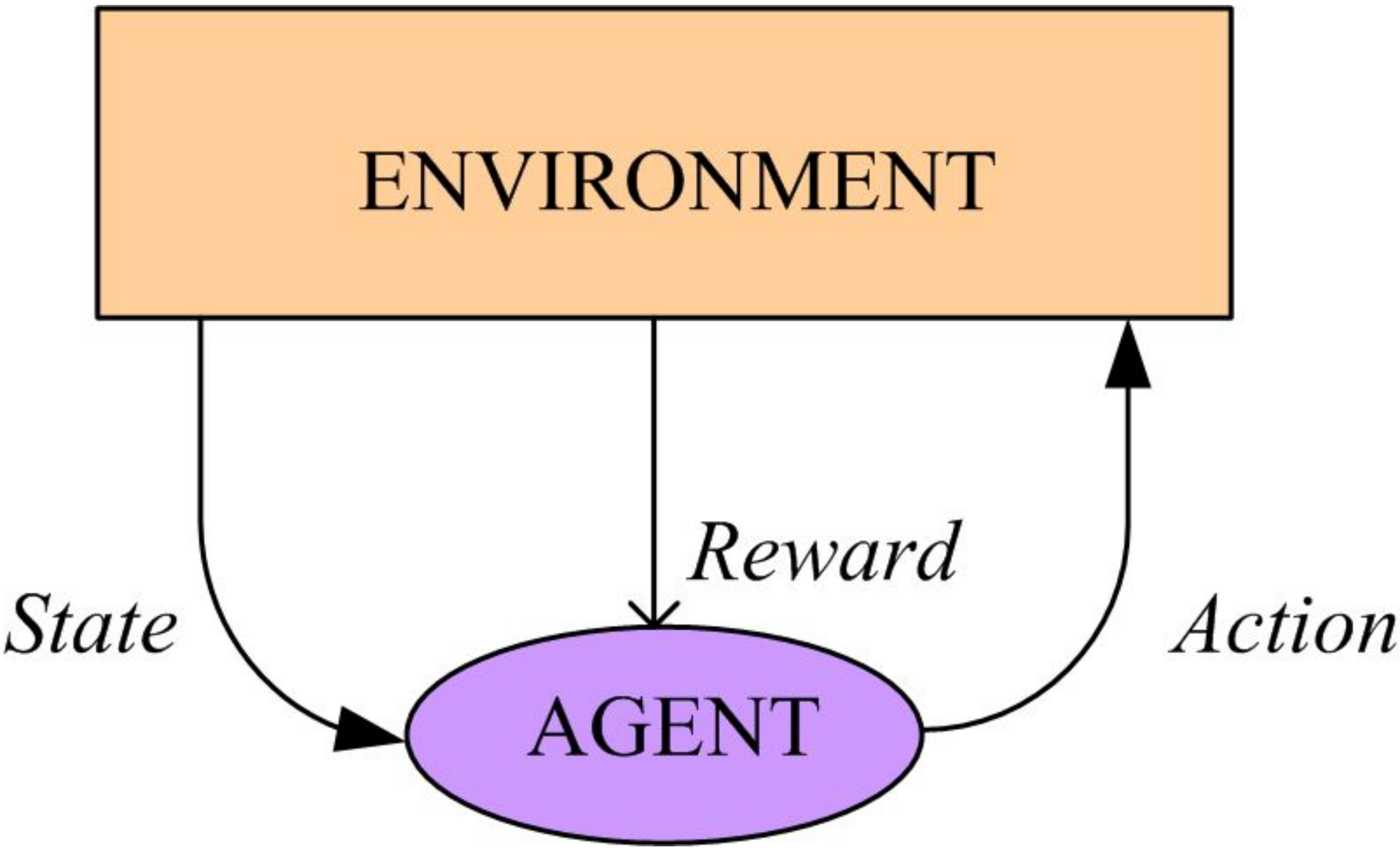




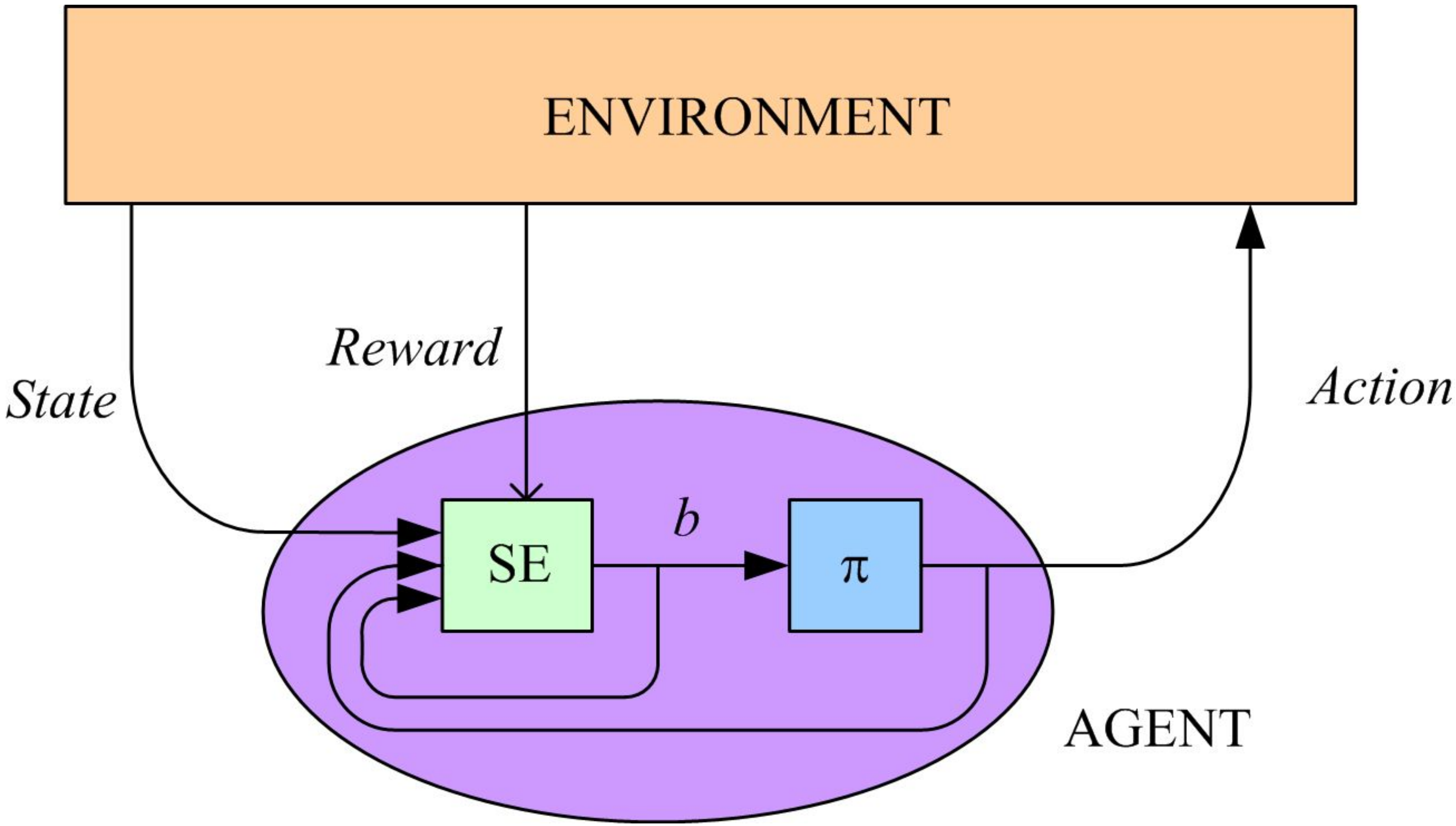
Reinforcement Learning

- Learning a policy: A sequence of outputs
- No supervised output but delayed reward
- Credit assignment problem
- Game playing
- Robot in a maze
- Multiple agents, partial observability, ...

Reinforcement Learning

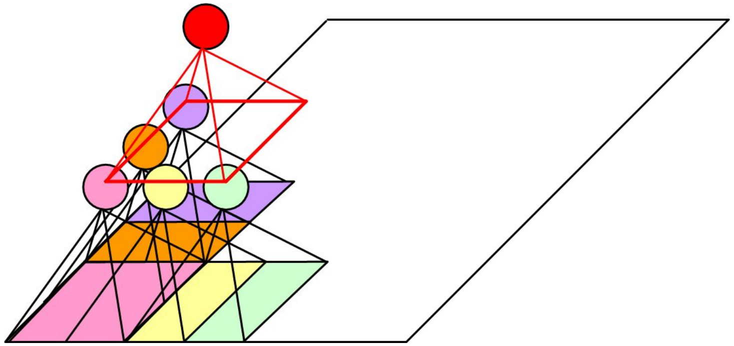


Reinforcement Learning



Summary

- Different types of learning
  - Association
  - Supervised Learning
    - Classification
    - Regression
  - Unsupervised Learning
  - Reinforcement Learning
- Biologically plausible machine learning?



McCulloch and Pitts neurons

A LOGICAL CALCULUS OF THE IDEAS IMMANENT IN NERVOUS ACTIVITY

WARREN S. MCCULLOCH and WALTER H. PITTS

Because of the "all-or-none" character of nervous activity, neural events and the relations among them can be treated by means of propositional logic. It is found that the behavior of every net can be described in these terms, with the addition of more complicated logical means for nets containing circles; and that for any logical expression satisfying certain conditions, one can find a net behaving in the fashion it describes. It is shown that many particular choices among possible neurophysiological assumptions are equivalent, in the sense that for every net behaving under one assumption, there exists another net which behaves under the other and gives the same results, although perhaps not in the same time. Various applications of the calculus are discussed.

A Logical Calculus of Ideas Immanent in Nervous Activity

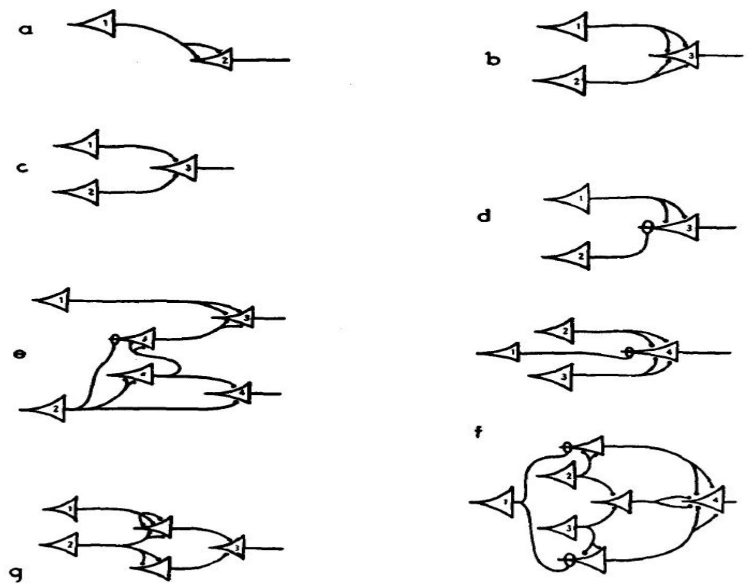


Figure 1a $N_2(t) . \equiv . N_1(t-1)$

Figure 1b $N_3(t) . \equiv . N_1(t-1) \vee N_2(t-1)$

Figure 1c $N_3(t) . \equiv . N_1(t-1) . N_2(t-1)$

Figure 1d $N_3(t) . \equiv . N_1(t-1) . \sim N_2(t-1)$

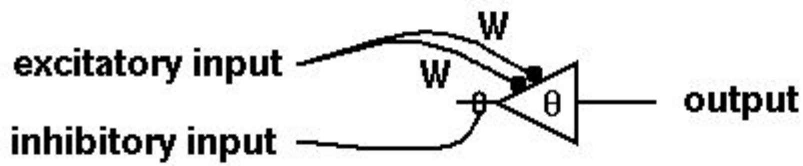
Figure 1e $N_3(t) :\equiv: N_1(t-1) . \vee . N_2(t-3) . \sim N_2(t-2)$
$N_4(t) . \equiv . N_2(t-2) . N_2(t-1)$

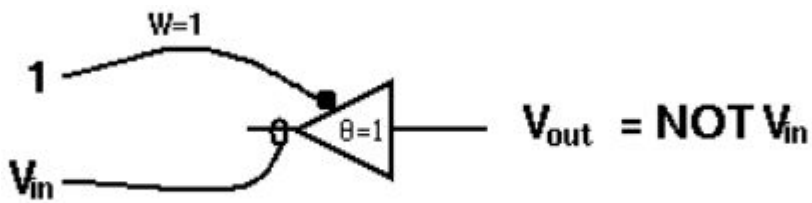
Figure 1f $N_4(t) :\equiv: \sim N_1(t-1) . N_2(t-1) \vee N_3(t-1) . \vee . N_1(t-1) . N_2(t-1) . N_3(t-1)$
$N_4(t) :\equiv: \sim N_1(t-2) . N_2(t-2) \vee N_3(t-2) . \vee . N_1(t-2) . N_2(t-2) . N_3(t-2)$

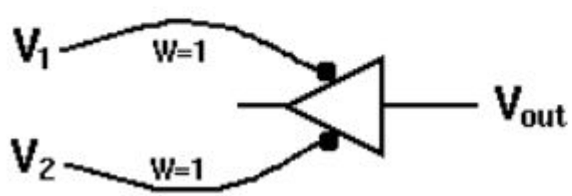
McCulloch and Pitts neurons

- McCulloch and Pitts (1943) assumptions:
- They are binary devices (Vi = [0,1])
- Each neuron has a fixed threshold, theta
- The neuron receives inputs from excitatory synapses, all having identical weights. (However it my receive multiple inputs from the same source, so the excitatory weights are effectively positive integers.)
- Inhibitory inputs have an absolute veto power over any excitatory inputs.
- At each time step the neurons are simultaneously (synchronously) updated by summing the weighted excitatory inputs and setting the output (Vi) to 1 iff the sum is greater than or equal to the threhold AND if the neuron receives no inhibitory input

McCulloch and Pitts neurons



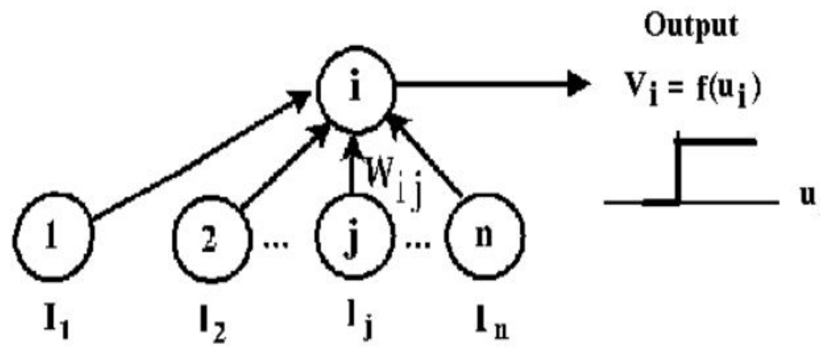




http://ecee.colorado.edu/~ecen4831/lectures/NNet2.html

Rosenblatt's simple perceptron

- The weights and thresholds were not all identical.
- Weights can be positive or negative.
- There is no absolute inhibitory synapse.
- Although the neurons were still two-state, the output function f(u) goes from [-1,1], not [0,1].
- Most importantly, there was a learning rule.



$$V_i = f(u_i) = \begin{cases} 0 & : \quad u_i < 0 \\ 1 & : \quad u_i \geq 0 \end{cases}$$

$$u_i = \sum_j W_{ij} I_j + \theta_i$$

http://ecee.colorado.edu/~ecen4831/lectures/NNet2.html

Learning with the perceptron

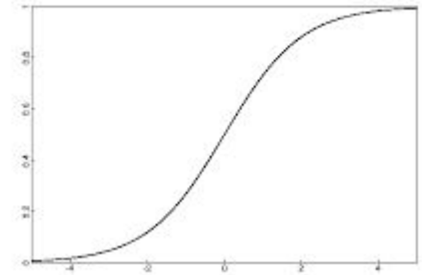
- $T = \{(\mathbf{x}_1, y_1), \dots (\mathbf{x}_n, y_n)\}$ is a training set of n pairs of input $\mathbf{x}_i$ and desired output y_i
- To learn the correct weights **w**:
  - Initialize **w** randomly
  - For each sample j do:
    - Calculate the actual output $y'_j = \mathbf{w}\mathbf{x}_j$
    - Adapt the weights $\mathbf{w}_k' = \mathbf{w}_k + \alpha(y_j - y_j')\mathbf{x}_{jk}$ for each $\mathbf{w}_k$
  - Repeat until the error is sufficiently small

Other considerations

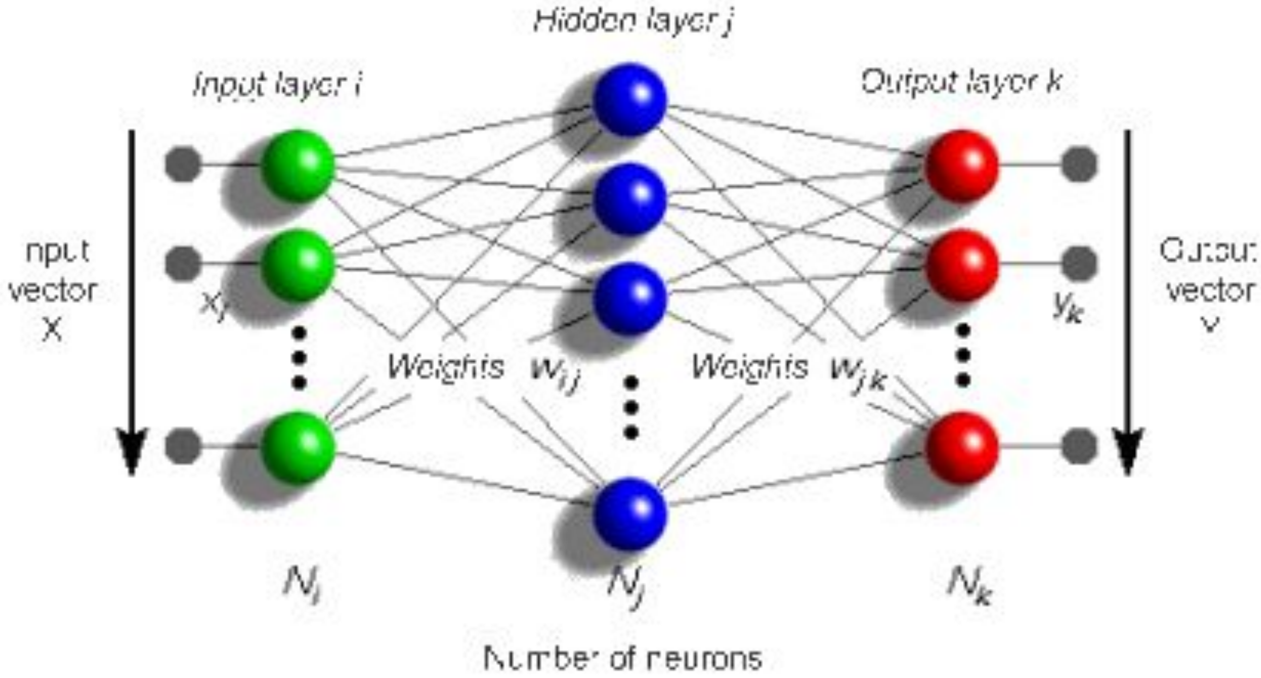
- Bias term: w_0
- Adding nonlinearity:
  - Logistic function
  - Hyperbolic tangent
- More complex networks

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

$$f(x) = \tanh(\beta x)$$

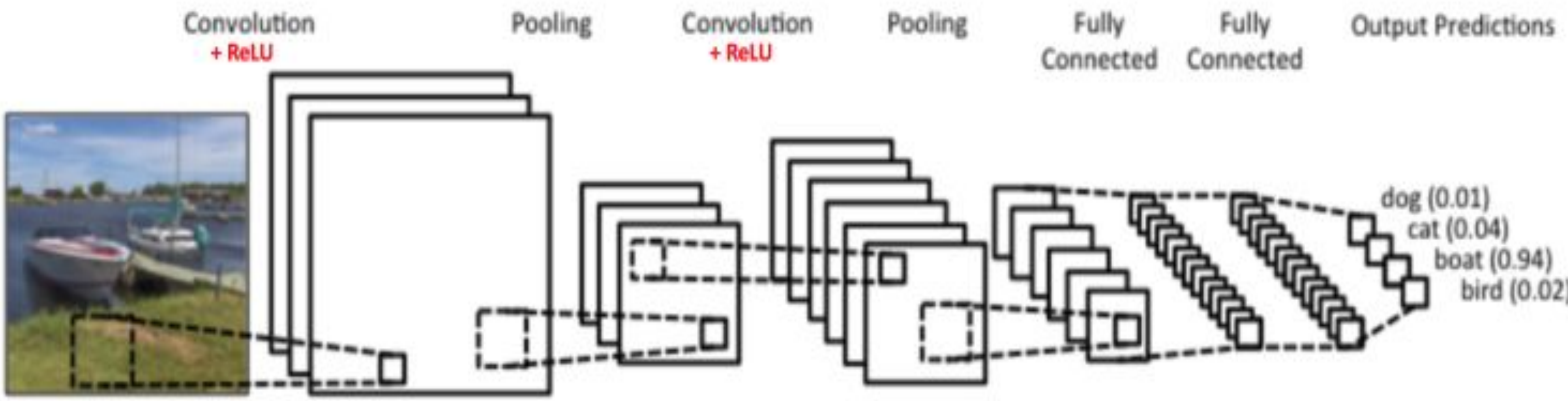


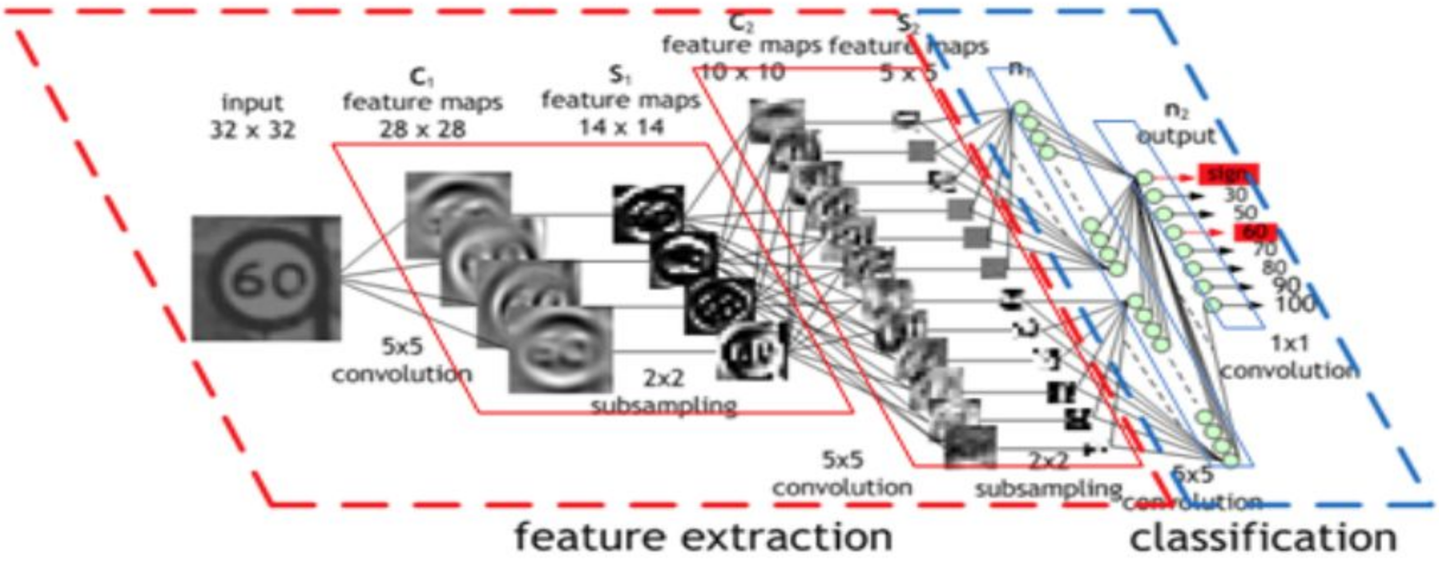
Multilayer perceptron



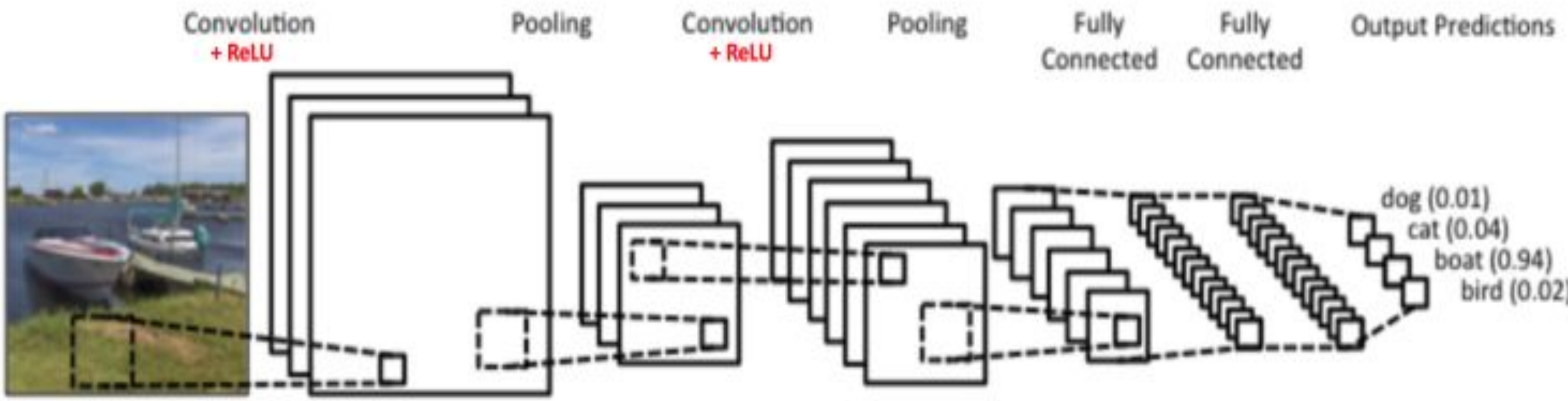
http://www.microstat-analytics.com/images/en/MLP.gif

Deep Neural Networks

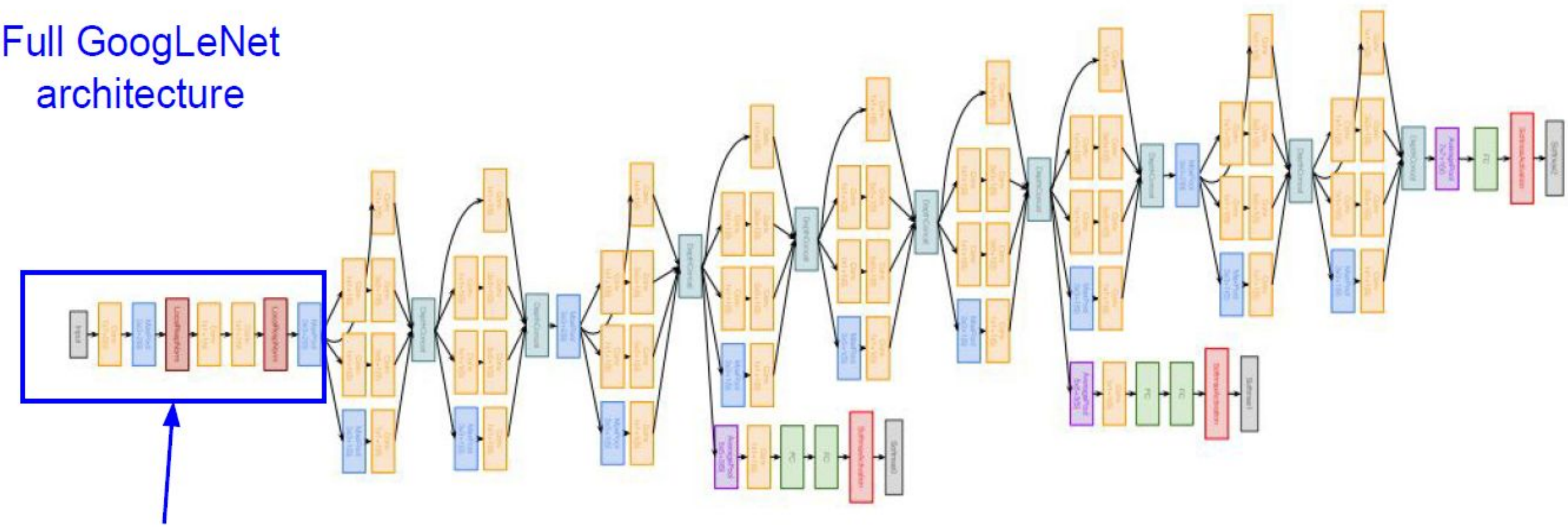




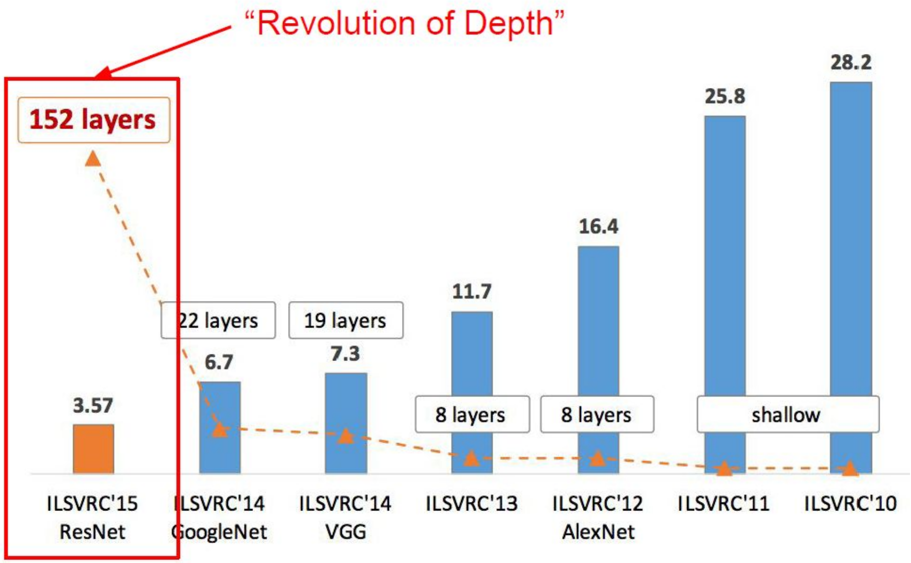
Deep Neural Networks

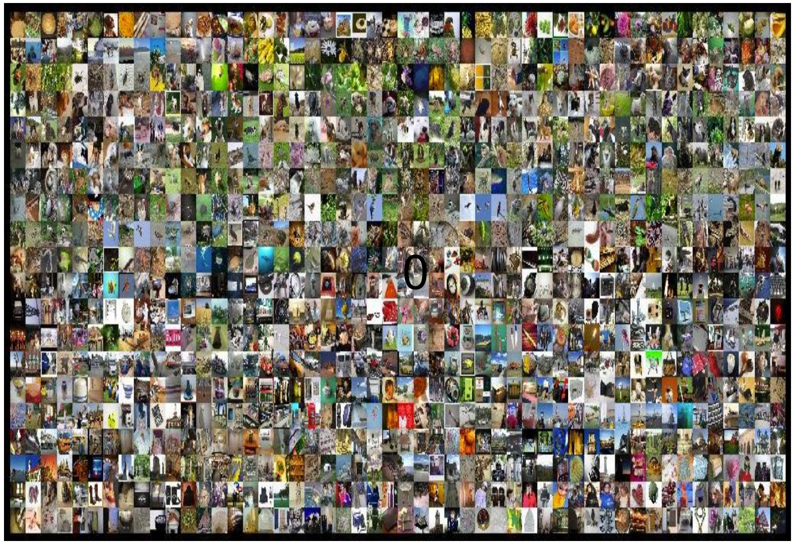


Full GoogLeNet
architecture



Deep Neural Networks





References

- Douglas Navarick – Learning and Memory.
- Matlin, Cognition, Chapter 8.
- E. Alpaydın, Intro. to Machine Learning