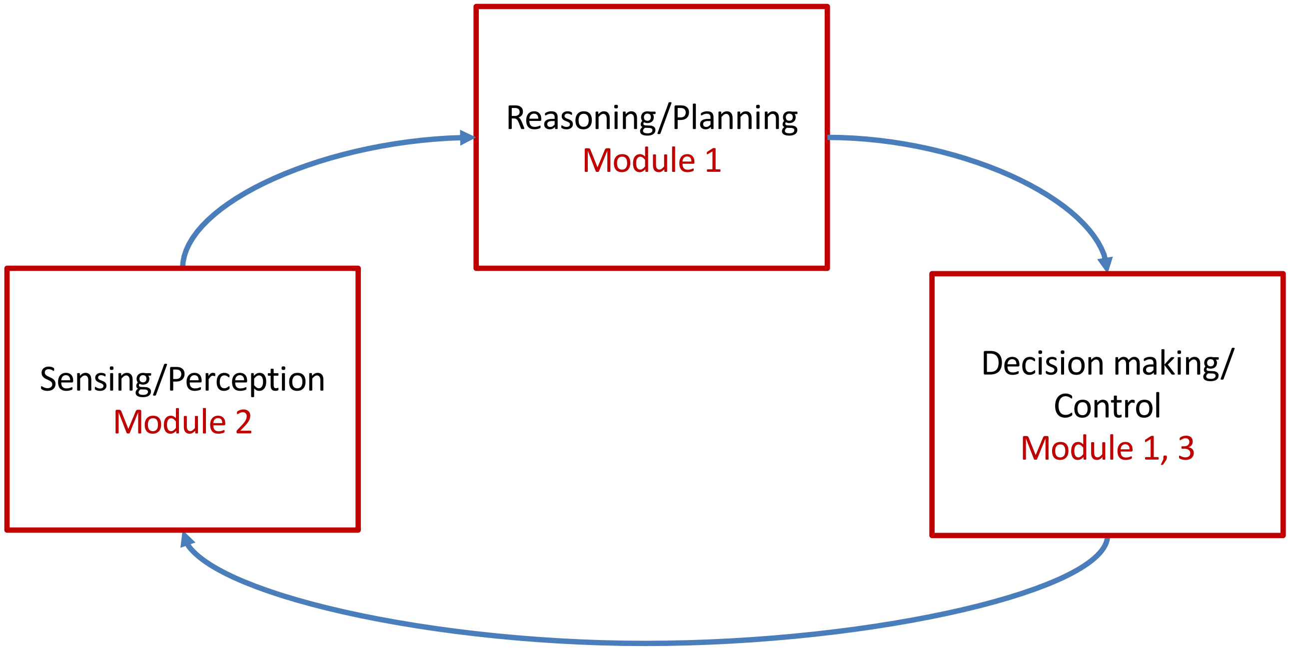
PERCEPTION/SENSING

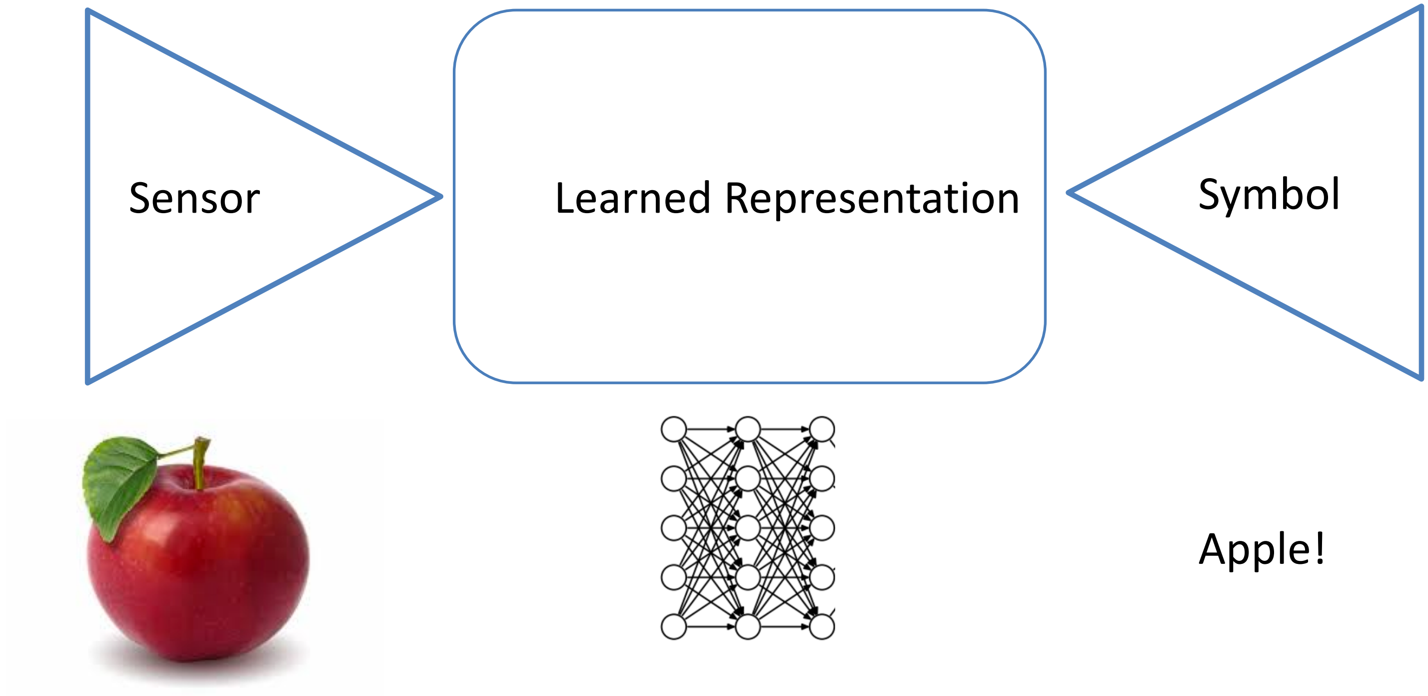
General autonomy triad



Symbols represent the world

- Agents/objects
  - State-space of objects
- Actions agents can take
  - Action space of agents
- Transition models
  - How do states of agents change when they take an action
- Reward models
  - What does the agent receive when it performs a task

Machine Learning and Perception



- Converting sensor information to symbols
- Uncovering transitions
- Modeling dependencies and correlations

Types of ML

- Regression

$$y = f(x)$$

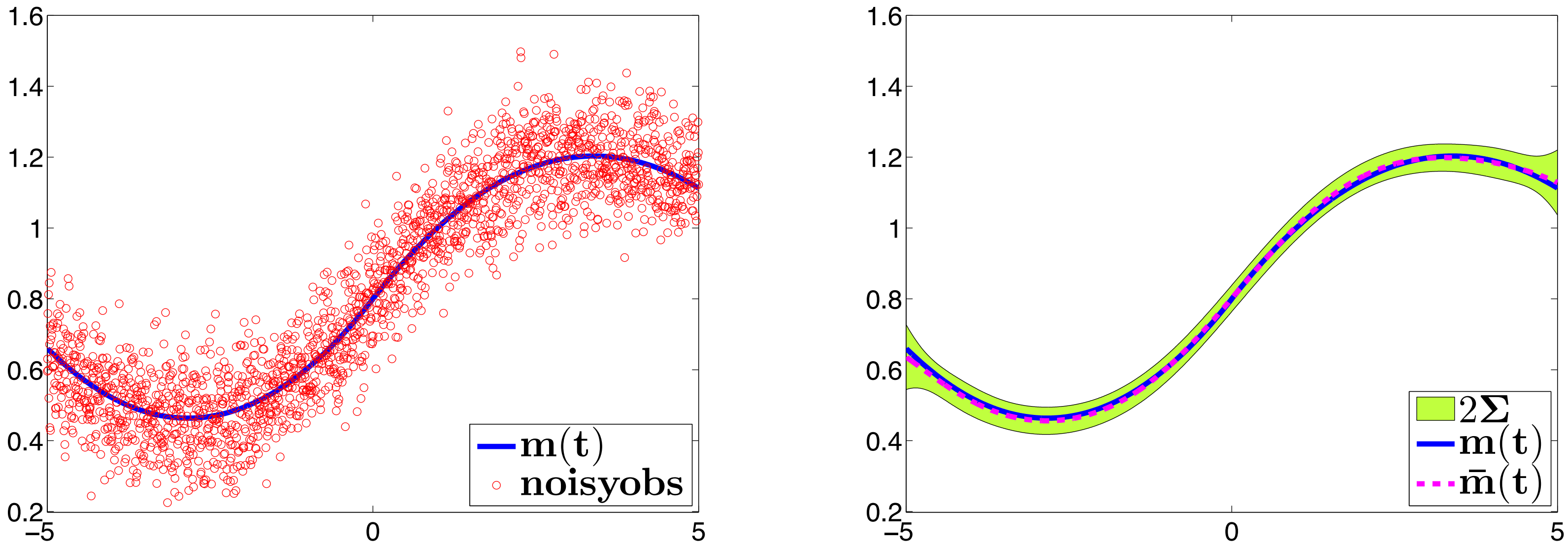
- Classification

$$p(y|x)$$

- Clustering

$$p_1(y \in Y_1), p_2(y \in Y_2), \ldots$$

Regression: Supervised learning



$$y = f(x)$$

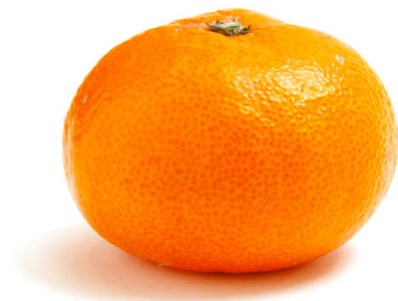
- Given y and x find f

Dynamic Learning: Transitions

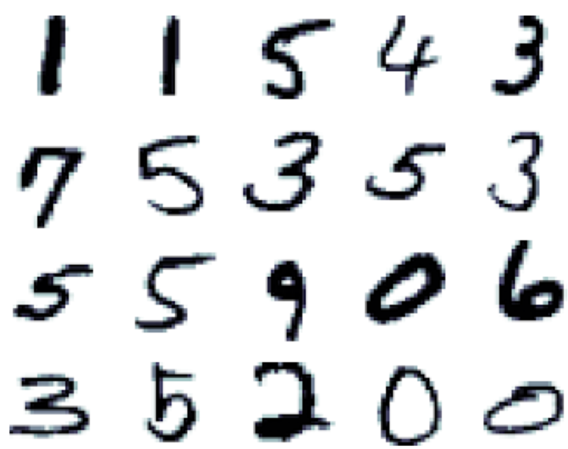
$$x_{t+1} = f(x_t)$$
$$y_t = g(x_t)$$

Classification: Supervised learning

$p(y|x)$



- Given x and y, find p



Clustering: Unsupervised learning

$p(y \in Y_1), p(y \in Y_2), \ldots$

Regression formalized

- Let $S = (x_1, x_2, x_3, \dots), Y = (y_1, y_2, y_3, \dots)$
- Consider a generalized linear model

$$y = w_1\phi_1(x) + w_2\phi_2(x) + w_3\phi_3(x) + \cdots w_n\phi_n(x)$$

- Rewriting

$$y = \boldsymbol{w}^T\boldsymbol{\phi}(x)$$

- Weight vector: $\boldsymbol{w} = [w_1, w_2, \dots, w_n]^T$,
- Feature vector: $\boldsymbol{\phi} = [\phi_1, \phi_2, \dots, \phi_n]^T$

Regression problem

- Minimize the regularized least squares cost

$$E(w) = \frac{1}{2}\sum_{n=1}^{N}\{y(x_n) - \boldsymbol{W}^T\boldsymbol{\phi}(x_n)\}^2 + \frac{1}{2}||\boldsymbol{w}||^2$$

Universal Approximation Theorem (Hornik NN; Park and Sandberg (RBFN)

$$||y(x) - \boldsymbol{W}^T\boldsymbol{\phi}(x)||_\infty \le \epsilon(x)$$

General Machine Learning Problem

- Find a feature vector and the weights such that a loss function is minimized

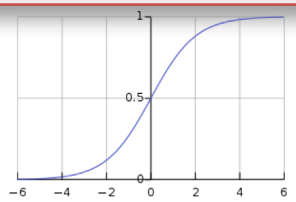
Optimization for regression

$$W = \arg\min_{W} L\big(y(x_n), \boldsymbol{W}^T \boldsymbol{\phi}(x_n)\big) \;\; for\ all\ x_n \in S$$

Logistic regression (classification)

$$\mu(x) = \sigma\big(\boldsymbol{W}^T \boldsymbol{\phi}(x)\big)$$

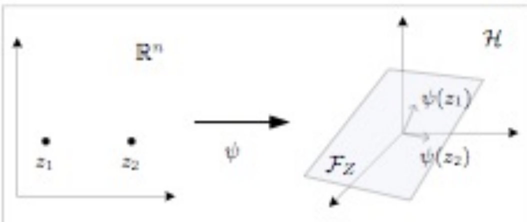
- σ is a squashing function: Sigmoidal, tanh, RELU etc.



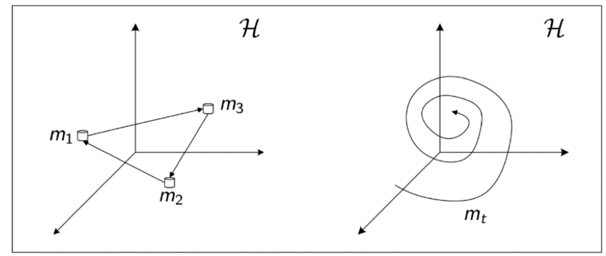
How do we find the features?

- Heuristics: Chosen from experience
  - Linear models
  - Polynomial models
  - Fourier series....
- Comparison
  - Kernel models
- Learned from data
  - Deep learning

Mercer Kernels



Hilbert Space mappings
Rassmussen, Schoelkopf…



Embedding dynamics in Hilbert Spaces
our work

- A Mercer kernel is a continuous positive definite function $k(x_1, x_2) \in R^+$ for $x_1, x_2 \in D$
- Example: Gaussian Radial Basis Functions: $\phi(x_1, x_2) = e^{-\frac{1}{\sigma^2}||x_1 - x_2||^2}$

Mercer: RKHS

There is higher (infinite) dimensional function space H, and a mapping $\psi: D \rightarrow \mathrm{H}$

$$k(x_1, x_2) =< \psi(x_1), \psi(x_2) >_H$$

Deep learning

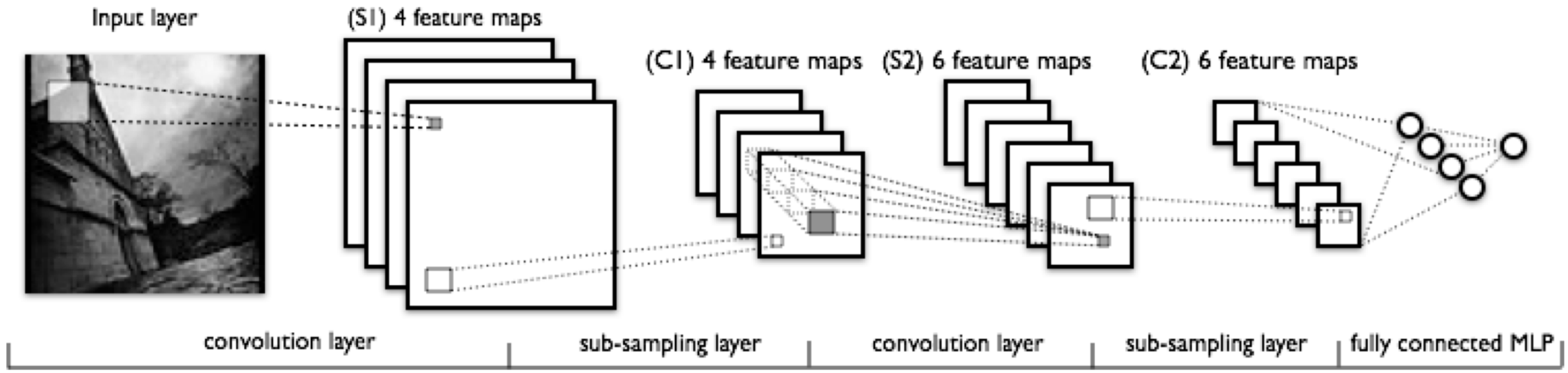
- Can we learn the feature vector?

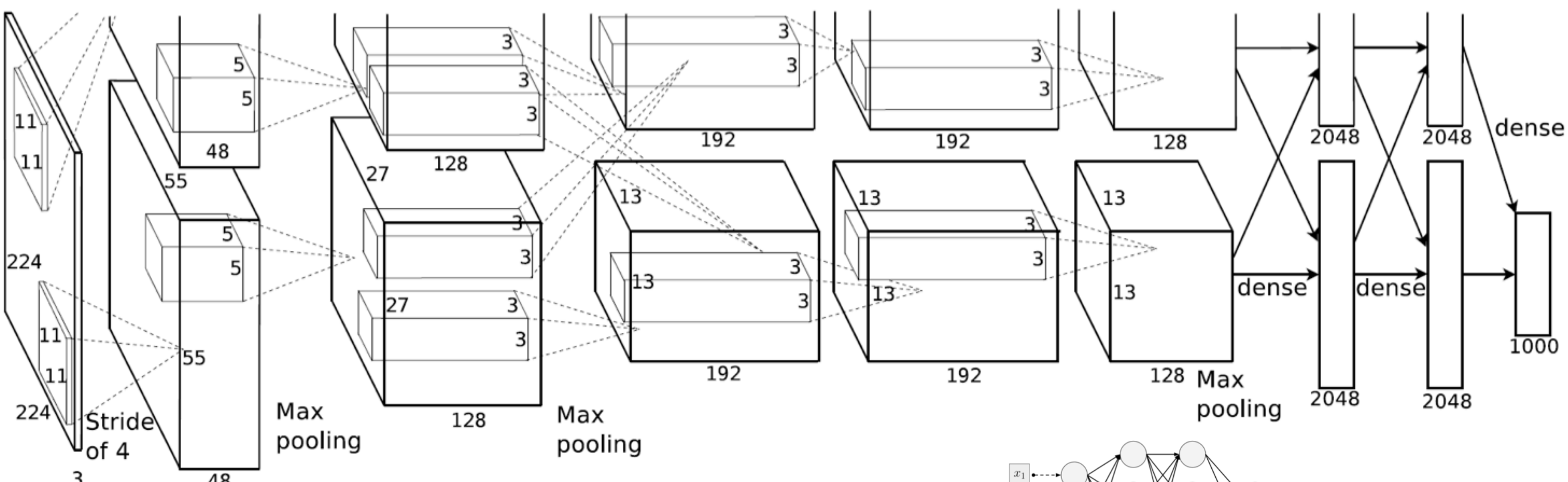
Model for learning the features

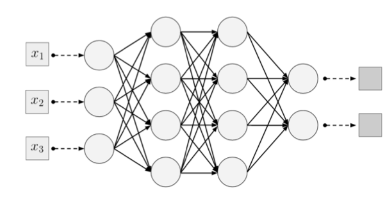
$$\phi(x) = \phi_1\left(w_1, y, \phi_2\left(w_2, \phi_3\left(\dots, \phi_n(w_n, x)\right) \dots\right)\right)$$

- More generally: functions can be learned as a directed acyclic graph
- Individual functions ϕ_i
  - Convolutions
  - Rectified linear units or other activation functions
  - Sigmoidal or other squashing functions
  - Maximum/average pooling functions

Deep Learning

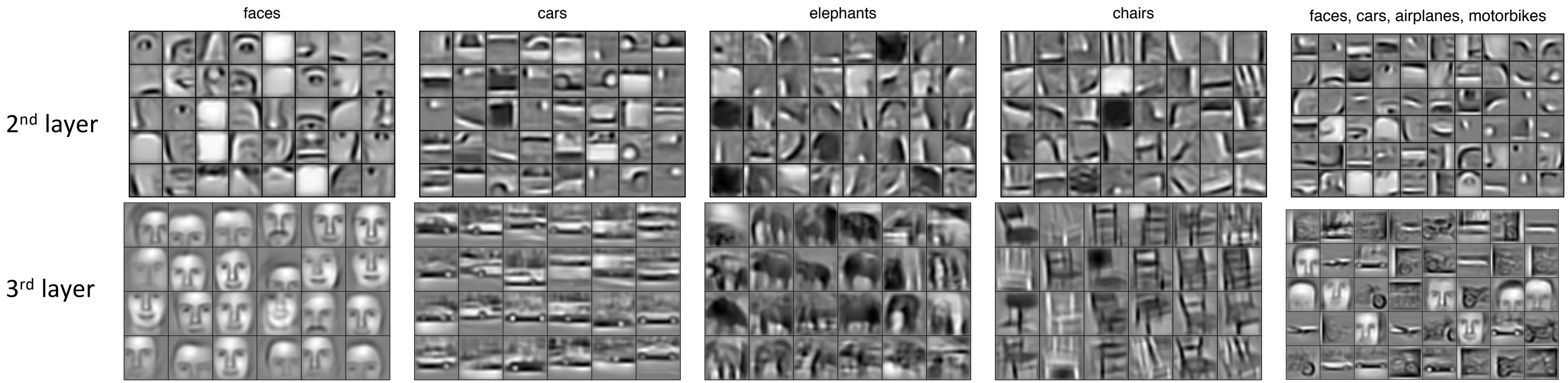


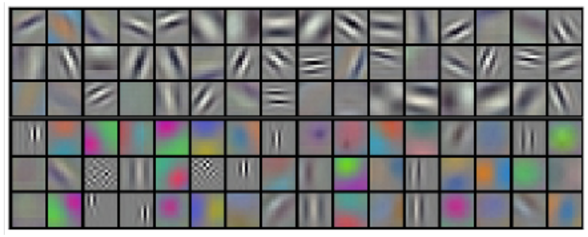


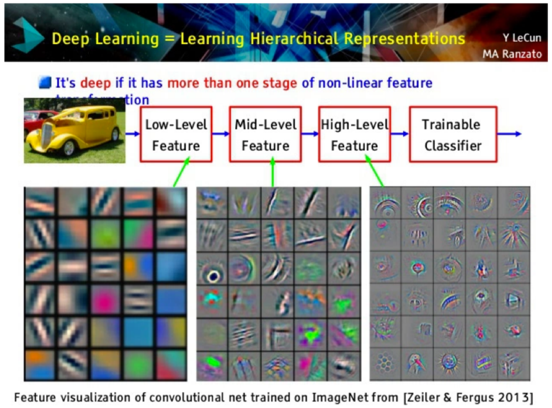


AlexNet

Learned features







- No need to heuristically find features, but needs lots of data and parallel computing