IntroML
ML. 1.2 Intro to unsupervised and reinforced learning

DataLab ICMAT

Intro to Unsupervised Learning

Elements of unsupervised learning

Given

- Input space $x \in \mathcal{X}$
- Training set $S = \{x_i\}_{i=1}^{N}$

Objective

- Learn model
- Infer some property $p(x)$
- Sample

Taxonomy of unsupervised learning algos

- Density estimation
- Manifold learning: PCA, non-linear PCA, …
- Finding modes and groups: cluster analysis, mixture models,…
- Sampling: GANs, Autoencoders, Variational autoencoders,…

Challenges in unsupervised learning

- High dimension of feature space
- Properties of interest more complex than parameter estimation
- No direct error quantification measure

Paradigm: Principal component analysis (PCA)

Two views

- Orthogonal projection to lower dimension space to maximize variance
- Linear projection minimizing average projection cost= average quadratic distance between data and projections

Applications

- Dimension reduction
- Compression
- Visualization
- Extraction of predictors. PC Regression
- ….

PCA: Maximum variance

Given

$$x_i \in \mathbb{R}^D \quad , \quad i = 1, \ldots, n$$

Find linear projection to space of smaller dimension maximizing variance of projected data

$$\Pi : \mathbb{R}^D \longrightarrow \mathbb{R}^M \quad , \quad M < D$$

PCA: Maximum variance

- 1 dimensional projection — $M=1,\ \mathbb{R}$
- Projection defined by — $u_1 \in \mathbb{R}^D \quad (u_1^t u_1 = 1)$
- Projection is — $u_1^t x$
- Mean of projected data — $\frac{1}{n}\sum_{i=1}^{n} u_1^t x = u_1^t \bar{x}$
- Variance of projected data — $\frac{1}{n}\sum_{i=1}^{n} \left(u_1^t x_i - u_1^t \bar{x}\right)^2 = u_1^t S u_1$

PCA: Maximum variance

- Problem to be solved

$$\max u_1^t S u_1 \quad \text{s.t. } u_1^t u_1 = 1$$

- Lagrangian formulation

$$\max_{u_1} u_1^t S u_1 + \lambda_1 (u_1^t u_1 - 1)$$

- Solution

$$S u_1 = \lambda_1 u_1$$
$$u_1^t S u_1 = \lambda_1$$

- Projection is eigenvector associated with first eigenvalue!!!

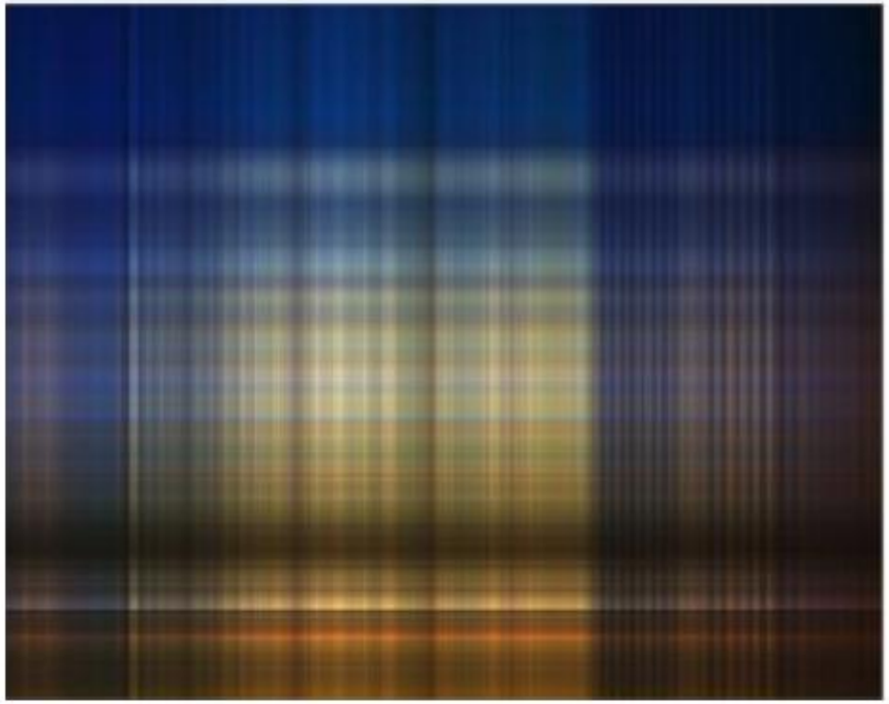
(and so on)

Data compression

Full example in Lab. Projecting each D-dimension point to M

$$\hat{x}_i = \bar{x} + \sum_{j=1}^{M} (x_i^t - \bar{x} u_j) u_j$$

- M = 1



- M = 3

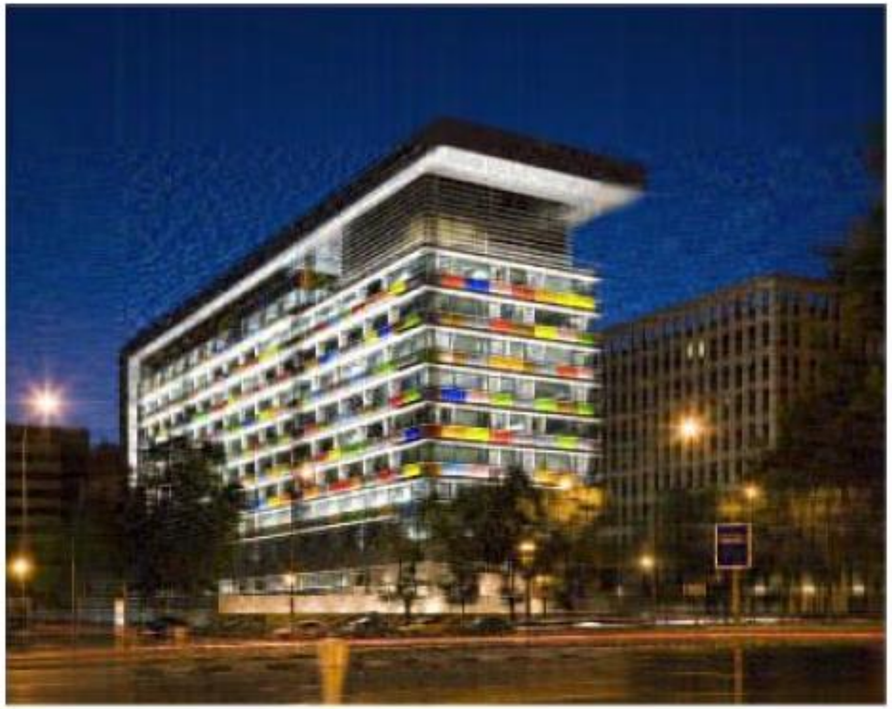
Data compression

- M = 10

- M = 20

Data compression

- M = 50



- M = 200

Implementation challenges

- High dimensionality. What if D>>n
- n points in space of dimension D
- Computational complexity of computing eigenvectors

Implementation challenges

A transformation that reduces complexity

$$X \in \mathbb{R}^{n \times D}, \quad x_i - \bar{x}$$

$$S = \frac{1}{n} X^t X \Rightarrow \frac{1}{n} X^t X u_j = \lambda_j u_j$$

$$\frac{1}{n} X X^t (X u_j) = \lambda_j (X u_j)$$

$$v_j = X u_j \rightarrow \left(\frac{1}{n} X X^t\right) v_j = \lambda_j v_j$$

$$O(D^3) \rightarrow O(N^3)$$

Implementation challenges

Transforming back

$$X^t \left(\frac{1}{n} X^t X\right)\underbrace{(X^t v_j)} = \lambda_j \underbrace{(X^t v_j)}$$

1. Compute eigenvectors
2. Transform back
3. Normalize

$$v_j$$
$$u_j = X^T v_j$$
$$u_j = \frac{1}{\sqrt{n\lambda_j}} X^T v_j$$

UL. To be seen

Deep neural nets (autoencoders, GANS)

UL. Advanced topics in PCA, matrix factorization, mixture models

Reinforcement learning

RL: features

- Learning by interaction with environment
  - ‘Cause-Effect’ relations
  - Consequences of actions
  - What to do to achieve goals
- Goal directed learning: what to do to maximize a reward
  - Discover actions that yield most reward by trying them (trial and error search)
  - Actions affect not only immediate reward but also affect environment (delayed reward)

RL features

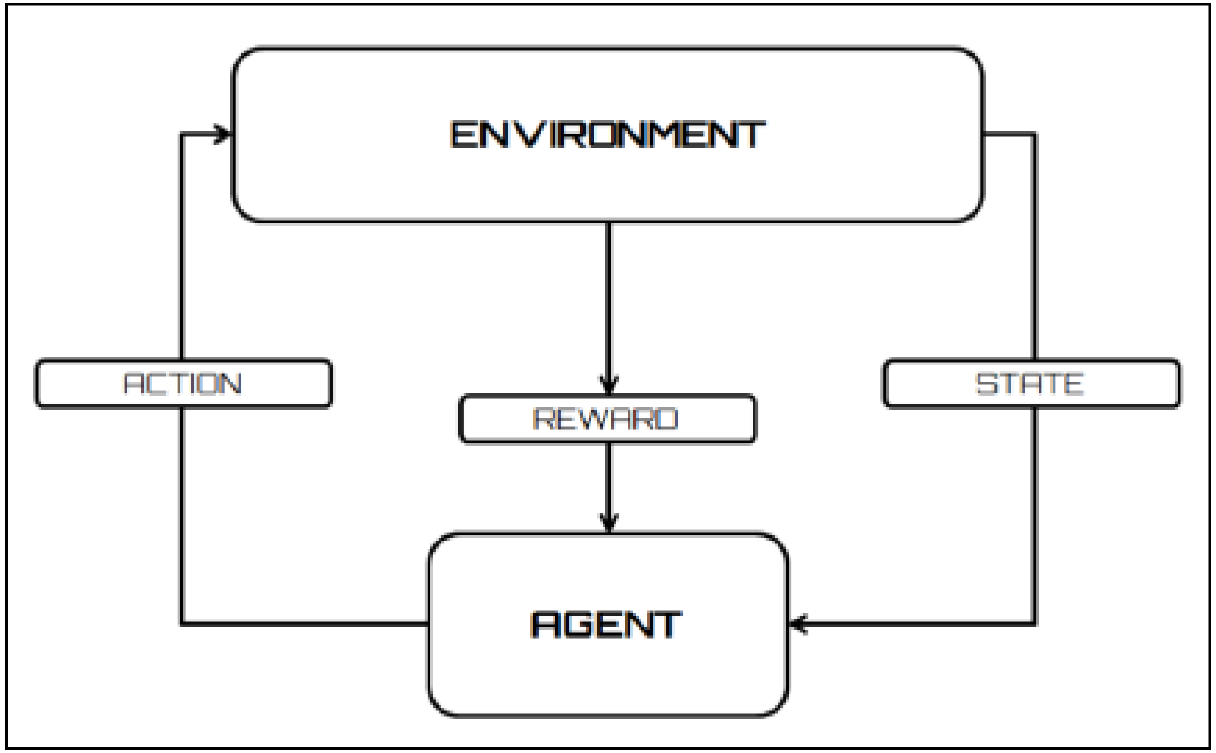
- Optimal control of incompletely known Markov decision processes
  - Schemes for sense-act-respond
  - Exploration (collect more info)-exploitation (best action)
  - Uncertainty about evolution of environment and rewards achieved
  - Sequential learning

RL elements

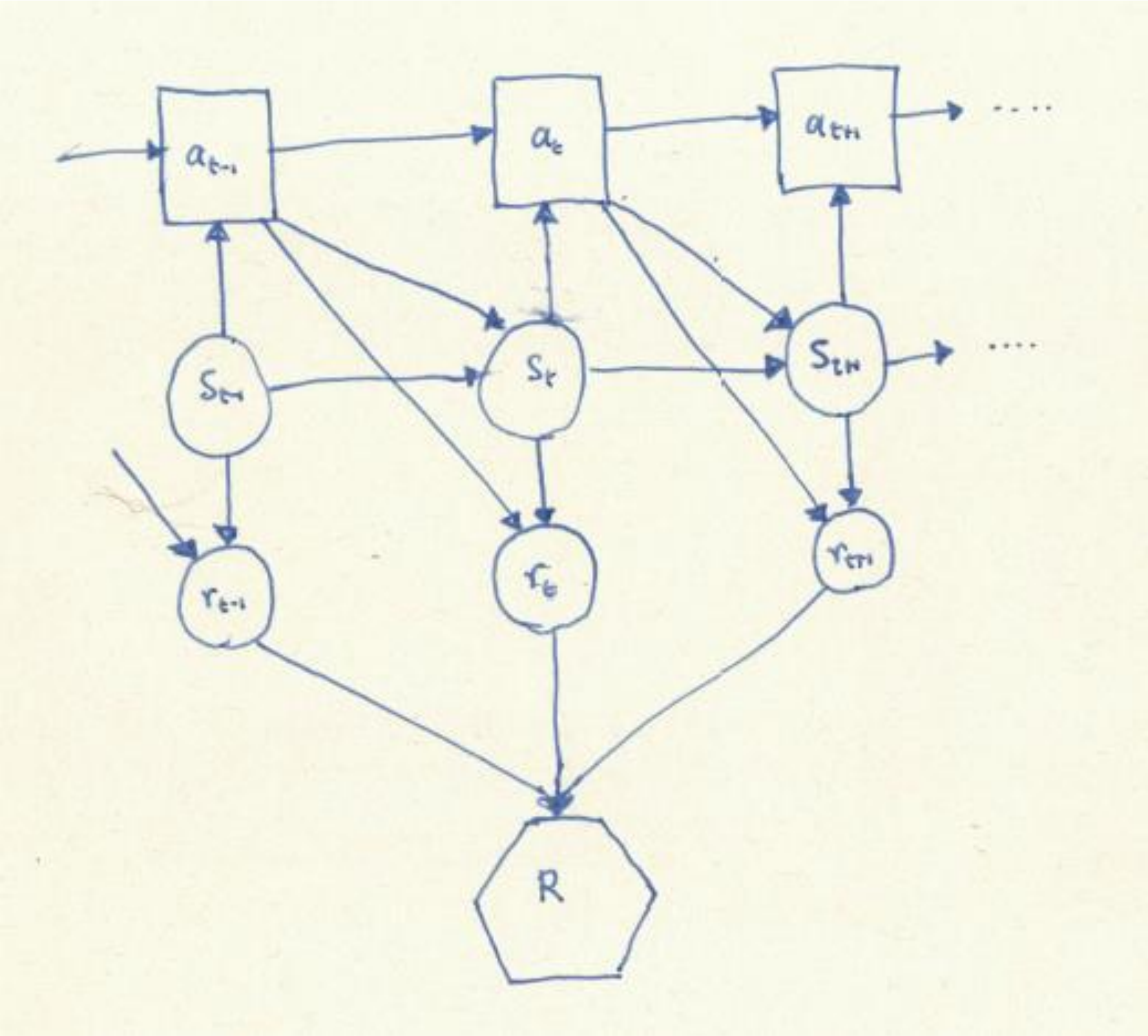
- Agent
- Environment with states
- Policy
- Reward signal
- Value function
- Model of environment

- Model based methods
- Model free methods

RL elements



RL elements



RL Elements: MDPs

- States
- Actions
- Transition
- Reward
- History
- LT Expected discounted utility
- Policy

$s \in S$

$a \in A$

$T : S \times A \rightarrow \Delta(S)$

$R : S \times A \rightarrow \Delta(R)$

$\tau = (s_0, a_0, s_1, a_1, \dots)$

$E_\tau \left(\sum_{t=0}^{\infty} \gamma^t R(a_t, s_t) \right)$

$\pi : S \rightarrow \Delta(A)$

RL elements: Q-learning

$$Q(s,a) := (1-\alpha)\, Q(s,a) + \alpha \left(r(s,a) + \gamma \max_{a'} Q(s', a') \right)$$

RL. To be seen

RL. MDPS, Dynamic programming, Q learning, policy gradient methods

Deep reinforcement learning