Introduction to Learning

Winter School ROBOTICA PRINCIPIA

Frederic Precioso
24/01/2019
Disclaimer

If any content in this presentation is yours but is not correctly referenced or if it should be removed, please just let me know and I will correct it.
Overview

• Context & Vocabulary
  – What represents Artificial Intelligence?
  – Machine Learning vs Data Mining?
  – Machine Learning vs Data Science?
  – Machine Learning vs Statistics?

• Unsupervised classification
• Explicit supervised classification
• Implicit supervised classification
• Deep Learning
• Reinforcement Learning
CONTEXT & VOCABULARY
WHAT REPRESENTS ARTIFICIAL INTELLIGENCE?
The term **Artificial Intelligence**, as a research field, was coined at the conference on the campus of Dartmouth College in the summer of **1956**, even though the idea was around since antiquity.

For instance in the first manifesto of Artificial Intelligence, “*Intelligent Machinery*”, in **1948** Alan Turing distinguished two different approaches to AI, which may be termed "*top-down*“ or **knowledge-driven AI** and "*bottom-up*“ or **data-driven AI**.

What is Artificial intelligence?

• The two different approaches to AI can be detailed:
  
  – "top-down“ or knowledge-driven AI
    • cognition = high-level phenomenon, independent of low-level details of implementation mechanism, first neuron (1943), first neural network machine (1950), neucognitron (1975)
  
  – "bottom-up“ or data-driven AI
    • opposite approach, start from data to build incrementally and mathematically mechanisms taking decisions

What is Artificial intelligence?

- *AI is originally defined, by Marvin Lee Minsky, as “the construction of computer programs doing tasks, that are, for the moment, accomplished more satisfyingly by human beings because they require high level mental processes such as: learning, perceptual organization of memory and critical reasoning”.*

- There are so the "artificial" side with the usage of computers or sophisticated electronic processes and the side “intelligence” associated with its goal to imitate the (human) behavior.

What is Artificial intelligence?

- The concept of **strong artificial intelligence** makes reference to a machine capable not only of producing intelligent behavior, but also to experience a feeling of a real sense of itself, “real feelings” (whatever may be put behind these words), and "an understanding of its own arguments”.

- The notion of **weak artificial intelligence** is a pragmatic approach of engineers: targeting to build more autonomous systems (to reduce the cost of their supervision), algorithms capable of solving problems of a certain class, etc. But this time, the machine *simulates* the intelligence, it seems to act as if it was smart.

Why Artificial Intelligence is so difficult to grasp?

• Frequently, when a technique reaches mainstream use, it is no longer considered as artificial intelligence; this phenomenon is described as the AI effect: "AI is whatever hasn't been done yet." (Larry Tesler's Theorem) -> e.g. Path Finding (GPS), Checkers game, Chess electronic game, Alpha Go...

⇒ “AI” is continuously evolving and so very difficult to grasp.
Machine Learning

\[
\begin{align*}
\begin{bmatrix} x \end{bmatrix} & \xrightarrow{\ f(x,\alpha) \ } y \\
\hline 
\text{Face Detection} \quad & \quad \text{Betting on sports} \\
\text{Scores, ranking...} \quad & \\
\text{Speech Recognition} \quad & \\
\end{align*}
\]
Machine Learning

\[
\begin{pmatrix}
X
\end{pmatrix}
\xrightarrow{f(X,\alpha)} y
\]

Support Vector Machines
Random Forest
Artificial Neural Networks
MACHINE LEARNING VS DATA MINING?
Data Mining Workflow
Data Mining Workflow

Data warehouse → Selection → Data preprocessing → Transformation → Data Mining → Model Patterns → Validation → Knowledge

Mainly manual
Data Mining Workflow

- Filling missing values
- Dealing with outliers
- Sensor failures
- Data entry errors
- Duplicates
- ...

Preprocessing

Selection

Data warehouse

Data

Data Mining

Transformation

Model Patterns

Validation

Knowledge
Data Mining Workflow

- Aggregation (sum, average)
- Discretization
- Discrete attribute coding
- Text to numerical attribute
- Scale uniformisation or standardisation
- New variable construction
- ...

- Selection
- Preprocessing
- Transformation
- Model Patterns
- Validation

Data Mining Workflow Diagram
Data Mining Workflow

- Regression
- (Supervised) Classification
- Clustering (Unsupervised Classification)
- Feature Selection
- Association analysis
- Novelty/Drift
- ...

Data Mining Workflow Diagram:
- Data warehouse
- Selection
- Preprocessing
- Transformation
- Data Mining
- Model Patterns
- Validation

Knowledge Output
Data Mining Workflow

- Data warehouse
- Selection
- Preprocessing
- Transformation
- Data Mining
- Model Patterns
- Validation

- Evaluation on Validation Set
- Evaluation Measures
- Visualization
- ...

Data Mining Workflow Diagram

Data → Selection → Preprocessing → Transformation → Data Mining → Model Patterns → Validation → Data
Data Mining Workflow

- Selection
- Preprocessing
- Transformation
- Data Mining
- Validation
- Model Patterns

- Data

- Visualization
- Reporting
- Knowledge
- ...

Data warehouse
Data Mining Workflow

**Problems**
- Regression
- (Supervised) Classification
- Density Estimation / Clustering (Unsupervised Classification)
- Feature Selection
- Association analysis
- Anomaly/Novelty/Drift
- ...

**Possible Solutions**
- Machine Learning
  - Support Vector Machine
  - Artificial Neural Network
  - Boosting
  - Decision Tree
  - Random Forest
  - ...
- Statistical Learning
  - Gaussian Models (GMM)
  - Naïve Bayes
  - Gaussian processes
  - ...
- Other techniques
  - Galois Lattice
  - ...

univ-cotedazur.fr
MACHINE LEARNING VS DATA SCIENCE?
Data Science Stack

Visualization / Reporting / Knowledge
- Dashboard (Kibana / Datameer)
- Maps (InstantAtlas, Leaflet, CartoDB...)
- Charts (GoogleCharts, Charts.js...)
- D3.js / Tableau / Flame

Analysis / Statistics / Artificial Intelligence
- Machine Learning (Scikit Learn, Mahout, Spark)
- Search / retrieval (ElasticSearch, Solr)

Storage / Access / Exploitation
- File System (HDFS, GGFS, Cassandra...)
- Access (Hadoop / Spark / Both, Sqoop...)
- Databases / Indexing (SQL / NoSQL / Both..., MongoDB, HBase, Infinispan)
- Exploit (LogStash, Flume...)

Infrastructures
- Grid Computing / HPC
- Cloud / Virtualization
MACHINE LEARNING VS STATISTICS?
What breed is that Dogmatix (Idéfix) ?

The illustrations of the slides in this section come from the blog “Bayesian Vitalstatistix: What Breed of Dog was Dogmatix?”
Does any real dog get this height and weight?

- Let us consider $\mathbf{x}$, vectors independently generated in $\mathbb{R}^d$ (here $\mathbb{R}^2$), following a probability distribution fixed but unknown $P(\mathbf{x})$. 
What should be the breed of these dogs?

- An Oracle assigns a value $y$ to each vector $x$ following a probability distribution $P(y|x)$ also fixed but *unknown*. 
An oracle provides me with examples?

- Let $S$ be a training set $S = \{(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)\}$, with $m$ training samples i.i.d. which follow the joint probability $P(x, y) = P(x)P(y|x)$. 
Statistical solution: Models, Hypotheses...
Statistical solution $P(\text{height, weight} \mid \text{breed})$...
Statistical solution $P(\text{height, weight} | \text{breed})$. ...
Statistical solution $P(\text{height, weight}|\text{breed})$...
Statistical solution: Bayes, \( P(\text{breed} | \text{height, weight}) \)...
Machine Learning

• we have a learning machine which can provide a family of functions \( \{f(x;\alpha)\} \), where \( \alpha \) is a set of parameters.

\[
\begin{pmatrix}
X \\
\end{pmatrix} \xrightarrow{f(X,\alpha)} \ y
\]
The problem in Machine Learning

- The problem of learning consists in finding the model \( \{f(x; \alpha)\} \) which provides the best approximation \( \hat{y} \) of the true label \( y \) given by the Oracle.
- **best** is defined in terms of minimizing a specific (error) cost related to your problem/objectives
  \[ Q((x, y), \alpha) \in [a; b]. \]
- Examples of cost/loss functions Q: Hinge Loss, Quadratic Loss, Cross-Entropy Loss, Logistic Loss...
Loss in Machine Learning

• **How to define the loss $L$ (or the cost $Q$)?**

You should choose the right loss function based on your problem and your data (here $y$ is the true/expected answer, $f(x)$ the answer predicted by the network).

**Classification**
- **Cross-entropy loss**: $L(x) = -(y \ln(f(x)) + (1-y)\ln(1-f(x)))$
- **Hinge Loss** (i.e. max-margin loss, i.e. 0-1 loss): $L(x) = \max(0, 1-yf(x))$
- ...

**Regression**
- **Mean Square Error** (or Quadratic Loss): $L(x) = (f(x)-y)^2$
- **Mean Absolute Loss**: $L(x) = |f(x)-y|$
- ...

If the loss is minimized but accuracy is low, you should check the loss function. Maybe it is not the appropriate one for your task.
The problem in Machine Learning

For Clarity sake, let us note $z = (x, y)$.

Thus, the objective is to minimize the Risk, i.e. the expectation of the error cost:

$$R(\alpha) = \int Q(z, \alpha) dP(z)$$

where $P(z)$ is unknown.

The training set $S = \{z_i\}_{i=1,...,m}$ is built through an i.i.d. sampling according to $P(z)$. Since we cannot compute $R(\alpha)$, we look for minimizing the Empirical Risk instead:

$$R_{emp}(\alpha) = \frac{1}{m} \sum Q(z_i, \alpha)$$
For Clarity sake, let us note $z = (x, y)$.

$S = \{z_i\}_{i=1,\ldots,m}$ is built through an \textit{i.i.d.} sampling according to $P(z)$.

\begin{align*}
\begin{array}{cc}
\textbf{Machine Learning} & \Longleftrightarrow \textbf{Statistics} \\
\text{Train through Cross-Validation} & \\
\end{array}
\end{align*}

\begin{align*}
\begin{array}{cc}
\textbf{Machine Learning} & \Longleftrightarrow \textbf{Statistics} \\
\text{Training set & Test set have to be distributed according to the same law (i.e. $P(z)$).} & \\
\end{array}
\end{align*}
Vapnik learning theory (1995)

Vapnik had proven the following equation \( \forall m \) with a probability at least equal to \( 1 - \eta \):

\[
R(\alpha_m) \leq R_{\text{emp}}(\alpha_m) + (b - a) \sqrt{\frac{d_{\text{VC}} \left( \ln \left( \frac{2m}{d_{\text{VC}}} \right) + 1 \right) - \ln(\eta/4)}{m}}
\]

Thus minimizing the \textbf{Risk} depends on minimizing the \textbf{Empirical Risk} and the \textit{confidence interval} which is linked to the term \( d_{\text{VC}} \) corresponding to the complexity of the model family chosen, i.e. the Vapnik-Chervonenkis dimension.
Vapnik learning theory (1995)

In his learning theory [Vapnik, 1995], Vapnik defines 4 fundamental steps:

- Study the theory of consistence of learning processes
- Define bounds on convergence speed of learning processes
- Handle the generalization power of learning processes
- Design a theory to build learning algorithms in order to find a tradeoff between minimizing the **Empirical Risk** and the **confidence interval** \(\Rightarrow\) minimization of the **Structural Risk**.
Machine Learning vs Statistics
UNSUPERVISED CLASSIFICATION
Unsupervised classification

- The system or the operator has only samples, but no label
- The number of classes and their nature have not been predetermined

⇒ unsupervised learning or clustering.
⇔ No expert is required.
⇔ The algorithm must discover by itself more or less hidden/underlying data structure.
Clustering: Partition a dataset into groups based on the similarity between the instances.
Clustering Algorithms (Partition)

- **Centroid-based (1957,1967)**
  (E.g. K-means, PAM, CLARA)

- **Hierarchical:**
  - Bottom up approach.
  - Top down approach.
Clustering Algorithms (Density)

- **Density-based**
  (E.g. DBSCAN, DENCLUE)

- **Distribution-based**
  (E.g. EM, extension of K-means)
Clustering Algorithms (Graph)

- **Graph-based**
  (E.g. Chameleon)

- **Grid-based**
  (E.g. STING, CLIQUE)
How many Clusters?
Which Algorithm?

K-means

PAM

Gaussian model

DBSCAN

Average linkage

DIANA
Algorithms’ Parameters

Avg. linkage: K=2
Avg. linkage: K=3
Avg. linkage: K=4
Avg. linkage: K=5
Avg. linkage: K=6
Avg. linkage: K=7
Which metric/distance?

Fig. 7. PAM clustering results of Scenario 2: (a) Euclidean distance ($K = 4$), (b) Manhattan distance ($K = 4$), (c) cosine distance ($K = 3$), (d) geodesic distance ($K = 8$, $K = 4$), and (e) density-based geodesic distance ($K = 30$, $K = 2$).
Clustering validation measures

- **Internal validation**: Separation? Compactness?
  E.g. Dunn, DB, and Silhouette indexes.

- **Problems**:
  - Different performance WRT existence of noise, variable densities, and non well-separated clusters.
  - Overrate the algorithm that uses the same clustering model.

- **External validation**: == Class labels?
  E.g. Rand, Jaccard, Purity, MI, VI indexes.

- **Problem**: Some class labels, at least, have to exist.
Consensus clustering

Ensemble of 3 base clusterings

Consensus solution
Overview

- Unsupervised classification
- Explicit supervised classification
  - Decision Trees
  - Random Forest
- Implicit supervised classification
- Deep Learning
- Reinforcement Learning
EXPLICIT SUPERVISED CLASSIFICATION
DECISION TREES
Decision tree to decide playing tennis or not

<table>
<thead>
<tr>
<th>outlook</th>
<th>temperature</th>
<th>humidity</th>
<th>windy</th>
<th>play</th>
</tr>
</thead>
<tbody>
<tr>
<td>sunny</td>
<td>hot</td>
<td>high</td>
<td>false</td>
<td>no</td>
</tr>
<tr>
<td>sunny</td>
<td>hot</td>
<td>high</td>
<td>true</td>
<td>no</td>
</tr>
<tr>
<td>overcast</td>
<td>hot</td>
<td>high</td>
<td>false</td>
<td>yes</td>
</tr>
<tr>
<td>rainy</td>
<td>mild</td>
<td>high</td>
<td>false</td>
<td>yes</td>
</tr>
<tr>
<td>rainy</td>
<td>cool</td>
<td>normal</td>
<td>false</td>
<td>yes</td>
</tr>
<tr>
<td>rainy</td>
<td>cool</td>
<td>normal</td>
<td>true</td>
<td>no</td>
</tr>
<tr>
<td>overcast</td>
<td>cool</td>
<td>normal</td>
<td>true</td>
<td>yes</td>
</tr>
<tr>
<td>sunny</td>
<td>mild</td>
<td>high</td>
<td>false</td>
<td>yes</td>
</tr>
<tr>
<td>sunny</td>
<td>cool</td>
<td>normal</td>
<td>false</td>
<td>yes</td>
</tr>
<tr>
<td>rainy</td>
<td>mild</td>
<td>normal</td>
<td>false</td>
<td>yes</td>
</tr>
<tr>
<td>sunny</td>
<td>mild</td>
<td>normal</td>
<td>true</td>
<td>yes</td>
</tr>
<tr>
<td>overcast</td>
<td>mild</td>
<td>high</td>
<td>true</td>
<td>yes</td>
</tr>
<tr>
<td>overcast</td>
<td>hot</td>
<td>normal</td>
<td>false</td>
<td>yes</td>
</tr>
<tr>
<td>rainy</td>
<td>mild</td>
<td>high</td>
<td>true</td>
<td>no</td>
</tr>
</tbody>
</table>

Objective
2 classes: yes & no
Prediction if a game will be played or not
Temperature will be easily converted into numerical

A simple example
• On the nodes
  – Distribution of the variable to predict

• The first node is segmented with the variable outlook (sunny, overcast, rainy): creation of 3 sub-groups
  – The first group contains 5 observations, 2 yes and 3 no

• The tree can be translated in a set of decision rules without loosing any information
  – Example: if outlook = sunny and humidity = high then play = yes
Basic algorithm

- **A** = BestAttribute(Examples) // Best attribute means more homogenous results
- Assign A to the root
- For each value of A, create a new sub-node of the roof
- Classify all the examples in the sub-nodes
- If all examples of a sub-node are **homogeneous**, assign their class to the node, if not repeat this process from this node

**Question: How to measure homogeneity?**
Entropy, Gini, Information Gain...
Decision tree to decide playing tennis or not
Final decision tree

- Outlook:
  - Sunny
  - Overcast
  - Rainy

- Humidity:
  - High
  - Normal

- Windy:
  - False
  - True

- Outcome:
  - No
  - Yes
  - Yes
  - No
Decision tree example

Note: tree boundaries are piecewise linear and axis-parallel
Advantages of decision trees

- Simple and easily interpretable rules (unlike implicit decision methods)
- No need to recode heterogeneous data
- Processing with missing values
- No model and no presupposition to meet (iterative method)
- Fast processing time
**Drawbacks**

- The nodes of level $n + 1$ are highly dependent on those of level $n$ (the modification of a single variable near to the top of the tree can entirely change the tree).
- We always choose the best local attributes, the best global information gain is not at all guaranteed.
- Learning requires a sufficient number of individuals.
- Inefficient when there are many classes.

- No convergence...
• We always chose the best local attributes, the best global information gain is not at all guaranteed

Solution chosen by the algorithm

Solution that should be chosen
Decision trees do not converge?

“Plant” a forest
Standard Random Forests

- Bagging
- Random Feature Selection
Error of generalization for Random Forest

- Error of generalization of RF can be bounded by:

\[ R(RF) \leq \frac{\rho(1 - s^2)}{s^2} \]

where
- \( \rho \) is the mean correlation between two decision trees
- \( s \) is the quality of prediction of the set of decision trees
Success story: Kinect

https://www.youtube.com/watch?v=IntbRsi8lU8
Success story: Kinect

Figure 3. Depth image features. The yellow crosses indicates the pixel x being classified. The red circles indicate the offset pixels as defined in Eq. 1. In (a), the two example features give a large depth difference response. In (b), the same two features at new image locations give a much smaller response.

Figure 4. Randomized Decision Forests. A forest is an ensemble of trees. Each tree consists of split nodes (blue) and leaf nodes (green). The red arrows indicate the different paths that might be taken by different trees for a particular input.
Success story: Kinect
Overview

• Unsupervised classification
• Explicit supervised classification
• Implicit supervised classification
  – Multi-Layer Perceptron
  – Support Vector Machine
• Deep Learning
• Reinforcement Learning
IMPLICIT SUPERVISED CLASSIFICATION
Thomas Cover’s Theorem (1965)  
“The Blessing of dimensionality”

Cover’s theorem states: A complex pattern-classification problem cast in a high-dimensional space non linearly is more likely to be linearly separable than in a low-dimensional space.

*(repeated sequence of Bernoulli trials)*
The curse of dimensionality [Bellman, 1956]

- Euclidian distance is not relevant in high dimension: $d \geq 10$
  1. look at the examples at distance at most $r$
  2. the hypersphere volume is too small: practically empty of examples

$$\frac{\text{volume of the sphere of radial } r}{\text{hypersphere of } 2r \text{ width}} \to d \to \infty 0$$

3. need a number of examples exponential in $d$

**Remark**

*Specific care for data representation*
MULTI-LAYER PERCEPTRON
First, biological neurons

Before we study artificial neurons, let’s look at a biological neuron.

Figure from K.Gurney, An Introduction to Neural Networks
First, biological neurons

Postsynaptic potential function with weight dependency, as a function of time (ms) and weight value, being excitatory in case of red and blue lines, and inhibitory in case of a green line.
Then, artificial neurons

Pitts & McCulloch (1943), binary inputs & activation function $f$ is a thresholding

Rosenblatt (1956), real inputs & activation function $f$ is a thresholding

(Schéma : Isaac Changhau)
Artificial neuron vs biology

Spike-based description

\[
\tau \frac{dV_m}{dt} = -V_m + I_{\text{inj}}
\]

Gradient descent: KO

Rate-based description

Steady regime

\[
y = s(\sum w_i x_i)
\]

Gradient descent: OK
From perceptron to network

@tachyeonz: A friendly introduction to neural networks and deep learning.
Single Perceptron Unit

- **Perceptron** only learns linear function [Minsky and Papert, 1969]

- Non-linear function needs layer(s) of neurons → Neural Network
- Neural Network = input layer + hidden layer(s) + output layer
Multi-Layer Perceptron

- Training a neural network [Rumelhart et al. / Yann Le Cun et al. 1985]
- **Unknown parameters: weights on the synapses**
- Minimizing a cost function: some metric between the predicted output and the given output

- Step function: non-continuous functions are replaced by a continuous non-linear ones
Multi-Layer Perceptron

- Minimizing a cost function: some metric between the predicted output and the given output

- Equation for a network of 3 neurons (i.e. 3 perceptrons):

\[ y = s(w_{13}s(w_{11}x_1 + w_{21}x_2 + w_{01}) + w_{23}s(w_{12}x_1 + w_{22}x_2 + w_{02}) + w_{03}) \]
Autonomous Land Vehicle In a Neural Network (ALVINN)

- ALVINN is an automatic steering system for a car based on input from a camera mounted on the vehicle.
  - Successfully demonstrated in a cross-country trip.
• The ALVINN neural network is:
  – 960 inputs (a 30x32 array derived from the pixels of an image),
  – 4 hidden units and
  – 30 output units (each representing a steering command).
Multi-Layer Perceptron

**Theorem [Cybenko, 1989]**

- A neural network with one single hidden layer is a **universal approximator**: it can represent any continuous function on compact subsets of $\mathbb{R}^n$.
- 2 layers is enough ... theoretically:
  “…networks with one internal layer and an arbitrary continuous sigmoidal function can approximate continuous functions with arbitrary precision providing that no constraints are placed on the number of nodes or the size of the weights".
- But **no efficient learning rule** is known and the size of the hidden layer is **exponential** with the complexity of the problem (which is unkown beforehand) to get an error $\varepsilon$, the layer must be infinite for an error $0$. 
SUPPORT VECTOR MACHINE

Cover’s theorem states: A complex pattern-classification problem cast in a high-dimensional space nonlinearly is more likely to be linearly separable than in a low-dimensional space.

*(repeated sequence of Bernoulli trials)*
The curse of dimensionality [Bellman, 1956]

- Euclidian distance is not relevant in high dimension: \( d \geq 10 \)

1. look at the examples at distance at most \( r \)
2. the hypersphere volume is too small: practically empty of examples

\[
\frac{\text{volume of the sphere of radial } r}{\text{hypersphere of } 2r \text{ width}} \to d \to \infty 0
\]

3. need a number of examples exponential in \( d \)

Remark

*Specific care for data representation*
SVM vs ANN

"SVMs have been developed in the reverse order to the development of neural networks (NNs). SVMs evolved from the sound theory to the implementation and experiments, while the NNs followed more heuristic path, from applications and extensive experimentation to the theory."

Support vector machines (SVMs) is a binary classification algorithm.

Extensions of the basic SVM algorithm can be applied to solve problems of regression, feature selection, novelty/outlier detection, and clustering.

SVMs are important because of (a) theoretical reasons:
- Robust to very large number of variables and small samples
- Can learn both simple and highly complex classification models
- Employ sophisticated mathematical principles to avoid overfitting

(b) superior empirical results.
Given training data:

\[
\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_N \in \mathbb{R}^n \\
y_1, y_2, \ldots, y_N \in \{-1, +1\}
\]

- Want to find a classifier (hyperplane) to separate negative objects from the positive ones.
- An infinite number of such hyperplanes exist.
- SVMs finds the hyperplane that maximizes the gap between data points on the boundaries (so-called “support vectors”).
- If the points on the boundaries are not informative (e.g., due to noise), SVMs may not do well.
Kernel Trick

https://www.youtube.com/watch?v=-Z4aojJ-pdg
Popular kernels

A kernel is a dot product in some feature space:

\[ K(\tilde{x}_i, \tilde{x}_j) = \Phi(\tilde{x}_i) \cdot \Phi(\tilde{x}_j) \]

**Examples:**

- Linear kernel:
  \[ K(\tilde{x}_i, \tilde{x}_j) = \tilde{x}_i \cdot \tilde{x}_j \]

- Gaussian kernel:
  \[ K(\tilde{x}_i, \tilde{x}_j) = \exp(-\gamma \| \tilde{x}_i - \tilde{x}_j \|^2) \]

- Exponential kernel:
  \[ K(\tilde{x}_i, \tilde{x}_j) = \exp(-\gamma \| \tilde{x}_i - \tilde{x}_j \|) \]

- Polynomial kernel:
  \[ K(\tilde{x}_i, \tilde{x}_j) = (p + \tilde{x}_i \cdot \tilde{x}_j)^q \]

- Hybrid kernel:
  \[ K(\tilde{x}_i, \tilde{x}_j) = (p + \tilde{x}_i \cdot \tilde{x}_j)^q \exp(-\gamma \| \tilde{x}_i - \tilde{x}_j \|^2) \]

- Sigmoidal:
  \[ K(\tilde{x}_i, \tilde{x}_j) = \tanh(k\tilde{x}_i \cdot \tilde{x}_j - \delta) \]
How to build a kernel function?

\[ k(x, y) = k_1(x, y) + k_2(x, y) \]

\[ k(x, y) = \alpha \cdot k_1(x, y) \]

\[ k(x, y) = k_1(x, y) \cdot k_2(x, y) \]

\[ k(x, y) = f(x) \cdot f(y) \]

with \( f() \) a function from input space to \( \mathbb{R} \)

\[ k(x, y) = k_3(\Phi(x), \Phi(y)) \]

\[ k(x, y) = xB y^T \]

with \( B \) a matrix \( N \times N \) symmetric, semi-definite positive
Complex kernels on Video Tubes

Video object extraction and description

- RoI = face tubes
- Frame face detection
- Face region grouping in shots

Example of a tube:
Complex kernels on Video Tubes

The major kernel on tubes is then defined as:

\[ K'_{\text{pow}}(T_i, T_j) = \left( \sum_r \sum_s \frac{|C_{ri}|}{\sqrt{|T_i|}} \frac{|C_{sj}|}{\sqrt{|T_j|}} k'(C_{ri}, C_{sj})^q \right)^{1/q} \]  \hspace{1cm} (1)

with the following minor kernel on chains:

\[ k'(C_{ri}, C_{sj}) = \exp \left( -\frac{1}{2\sigma^2} \chi^2 \left( \frac{\chi}{\sigma^2} \right) \right) e^{-\frac{(x_{ri} - x_{sj})^2 + (y_{ri} - y_{sj})^2}{2\sigma^2}} \]
Classification
Classification
SVM are ANN
Overview

- Unsupervised classification
- Explicit supervised classification
- Implicit supervised classification
- Deep Learning
  - Convolutional Neural Networks (CNN)
  - Generative Adversarial Networks (GAN)
  - Stacked Denoising AutoEncoder (SDAE)
  - Recurrent Neural Networks (RNN)
- Reinforcement Learning
DEEP LEARNING
Deep representation origins

- **Theorem Cybenko (1989)** A neural network with one single hidden layer is a universal “approximator”, it can represent any continuous function on compact subsets of $\mathbb{R}^n$ $\Rightarrow$ 2 layers are enough...but hidden layer size may be exponential
Deep representation origins

- **Theorem Hastad** (1986), **Bengio** et al. (2007) Functions representable compactly with $k$ layers may require exponentially size with $k-1$ layers
Deep representation origins

- **Theorem Hastad** (1986), **Bengio** et al. (2007) Functions representable compactly with $k$ layers may require exponentially size with $k-1$ layers.
Enabling factors

• Why do it now? Before 2006, training deep networks was unsuccessful because of practical aspects
  — faster CPU's
  — parallel CPU architectures
  — advent of GPU computing

• Results...
  — 2009, sound, interspeech + ~24%
  — 2011, text, + ~15% without linguistic at all
  — 2012, images, ImageNet + ~20%

Structure the network?

- Can we put any structure reducing the space of exploration and providing useful properties (invariance, robustness...)?

\[ y = s(w_{13}s(w_{11}x_1 + w_{21}x_2 + w_{01}) + w_{23}s(w_{12}x_1 + w_{22}x_2 + w_{02}) + w_{03}) \]
CONVOLUTIONAL NEURAL NETWORKS (AKA CNN, CONVNET)
Convolutional neural network

- Deep Networks are as good as humans at recognition, identification...

How much does a deep network understands those tasks?
Deep representation by CNN

The ventral (recognition) pathway in the visual cortex has multiple stages:
- Retina - LGN - V1 - V2 - V4 - PIT - AIT ....

Lots of intermediate representations

[picture from Simon Thorpe]

[Gallant & Van Essen]
Convolution in nature
Convolution

Center element of the kernel is placed over the source pixel. The source pixel is then replaced with a weighted sum of itself and nearby pixels.

\[
\begin{align*}
(4 \times 0) & \\
(3 \times 0) & \\
(2 \times 0) & \\
(1 \times 1) & \\
(0 \times 1) & \\
(0 \times 0) & \\
(0 \times -2) & \\
\end{align*}
\]

-8
1. Hubel and Wiesel have worked on visual cortex of cats (1962)
2. Convolution
3. Pooling
Convolution = Perceptron

Center element of the kernel is placed over the source pixel. The source pixel is then replaced with a weighted sum of itself and nearby pixels.

\[ y = \text{sign}(w \cdot x) \]

\[
\begin{vmatrix}
(4 \times 0) & (0 \times 0) & (0 \times 0) & (0 \times 0) \\
(0 \times 0) & (0 \times 0) & (0 \times 1) & (0 \times 1) \\
(0 \times 0) & (0 \times 0) & (0 \times 0) & (0 \times 1) \\
(4 \times 0) & (0 \times 0) & (0 \times 0) & (0 \times 0)
\end{vmatrix}
= -8
\]
Convolution = Perceptron

\[ y = \text{sign}(w \cdot x) \]

Center element of the kernel is placed over the source pixel. The source pixel is then replaced with a weighted sum of itself and nearby pixels.

\[
\begin{align*}
(4 \times 0) \\
(0 \times 0) \\
(0 \times 0) \\
(0 \times 0) \\
(0 \times 1) \\
(0 \times 1) \\
(0 \times 0) \\
(0 \times 1) \\
+ (-4 \times 2) \\
-8
\end{align*}
\]

Convolution kernel (emboss)

New pixel value (destination pixel)
If convolution = perceptron

1. Convolution

2. Pooling
Deep representation by CNN

- feature map = result of the convolution
- convolution with a filter extract characteristics (edge detectors)
- extract parallelised characteristics at each layer

1. final representation of our data
2. classifier (MLP)
Deep representation by CNN
Deep representation by CNN
Deep representation by CNN

Learning of object parts

Examples of learned object parts from object categories

Faces | Cars | Elephants | Chairs
--- | --- | --- | ---
Fig. 2: EndoNet architecture (best seen in color). The layers shown in the turquoise rectangle are the same as in the AlexNet architecture.
Endoscopic Vision Challenge 2017
Surgical Workflow Analysis in the SensorOR

10th of September
Quebec, Canada
Clinical context: Laparoscopic Surgery

Surgical Workflow Analysis
Task

Phase segmentation of laparoscopic surgeries

Video

Surgical Devices
Dataset

30 colorectal laparoscopies
- Complex type of operation
- Duration: 1.6h – 4.9h (avg 3.2h)
- 3 different sub-types
  - 10x Proctocolectomy
  - 10x Rectal resection
  - 10x Sigmoid resection

Sensor data recorded in integrated OR (Karl Storz OR1)
- Laparoscopic image stream
- Surgical devices
### Annotation

Annotated by surgical experts, 13 different phases

<table>
<thead>
<tr>
<th>Phase ID</th>
<th>Phase</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Preparation and orientation at abdomen</td>
</tr>
<tr>
<td>1</td>
<td>Dissection of lymphnodes and blood vessels</td>
</tr>
<tr>
<td>2</td>
<td>Retroperitoneal preparation to lower pancreatic border</td>
</tr>
<tr>
<td>3</td>
<td>Retroperitoneal preparation of duodenum and pancreatic head</td>
</tr>
<tr>
<td>4</td>
<td>Mobilizing the sigmoid and the descending colon</td>
</tr>
<tr>
<td>5</td>
<td>Mobilizing the splenic flexure</td>
</tr>
<tr>
<td>6</td>
<td>Mobilizing the tranverse colon</td>
</tr>
<tr>
<td>7</td>
<td>Mobilizing the ascending colon</td>
</tr>
<tr>
<td>8</td>
<td>Dissection and resection of rectum</td>
</tr>
<tr>
<td>9</td>
<td>Preparing the anastomosis extraabdominally</td>
</tr>
<tr>
<td>10</td>
<td>Preparing the anastomosis intraabdominally</td>
</tr>
<tr>
<td>11</td>
<td>Placing stoma</td>
</tr>
<tr>
<td>12</td>
<td>Finishing the operation</td>
</tr>
<tr>
<td>13</td>
<td>Exception (will be ignored during evaluation)</td>
</tr>
</tbody>
</table>
**Method**

**Spatial Network**

- **Temporal Network**
  - ResNet-34
  - 7x7 conv
  - 3x3 conv
  - 3x3 conv
  - 3x3 conv
  - feature vector FV (512)

**Final Network**

- Spatial CNN
- Temporal CNN
- Final Network Accuracy:
  - Rectal resection: 11
  - Sigmoid resection: 10
  - Proctolectomy: 12
  - Rectal resection: 62.91%
  - Sigmoid resection: 63.01%
  - Proctolectomy: 63.26%

**Number of target classes:**
- Rectal resection: 11
- Sigmoid resection: 10
- Proctolectomy: 12

**Temporal network accuracy:**
- Rectal resection: 49.88%
- Sigmoid resection: 48.56%
- Proctolectomy: 46.96%
And the winner is...

<table>
<thead>
<tr>
<th></th>
<th>Data used</th>
<th>Average Jaccard</th>
<th>Median Jaccard</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Video</td>
<td>40%</td>
<td>38%</td>
<td>61%</td>
</tr>
<tr>
<td>2</td>
<td>Video + Device</td>
<td>38%</td>
<td>38%</td>
<td>60%</td>
</tr>
<tr>
<td>3</td>
<td>Video</td>
<td>25%</td>
<td>25%</td>
<td>57%</td>
</tr>
<tr>
<td>4</td>
<td>Device</td>
<td>16%</td>
<td>16%</td>
<td>36%</td>
</tr>
<tr>
<td>5</td>
<td>Video</td>
<td>8%</td>
<td>7%</td>
<td>21%</td>
</tr>
</tbody>
</table>
Before Deep Learning

*Hand-crafted/Engineered features*

– Image recognition
  – Pixel → edge → texton → patterns → part → object

– Text
  – Character → word → word group → clause → sentence → story

– Speech
  – Sample → spectral band → sound → ... → phone → phoneme → word

Since Deep Learning

→ Hierarchy of representations with increasing level of abstraction
→ Each stage is a kind of trainable feature transform...

*as long as you have enough data to train the hierarchy*

Le Cun - Ranzato
How Deep Learning?

Start from raw data OR from a first level representation?

Tractable feature transform

Gray-level Pixels

Color Pixels

Text Embedding

Waves

Images

Decision

Le Cun - Ranzato
AMAZING BUT...
Man is to Computer Programmer as Woman is to Homemaker?
Debiasing Word Embeddings

Tolga Bolukbasi\textsuperscript{1}, Kai-Wei Chang\textsuperscript{2}, James Zou\textsuperscript{2}, Venkatesh Saligrama\textsuperscript{1,2}, Adam Kalai\textsuperscript{2}

\textsuperscript{1}Boston University, 8 Saint Mary’s Street, Boston, MA
\textsuperscript{2}Microsoft Research New England, 1 Memorial Drive, Cambridge, MA
tolgab@bu.edu, kw@kwchang.net, jamesyzou@gmail.com, srv@bu.edu, adam.kalai@microsoft.com

Abstract
Amazing but... be careful of the adversaries (as any other ML algorithms)
Amazing but... be careful of the adversaries (as any other ML algorithms)
Amazing but... be careful of the adversaries (as any other ML algorithms)
Amazing but...be careful of the adversaries (as any other ML algorithms)
Amazing but... be careful of the adversaries (as any other ML algorithms)
Amazing but... be careful of the adversaries

https://nicholas.carlini.com/code/audio_adversarial_examples/
GENERATIVE ADVERSARIAL NETWORKS
How to solve it?
Generative Adversarial Networks
It finally did not solve adversarial, but...

Operations between latent representations (manifold)
It finally did not solve adversarial, but...
(DENOISING) STACKED AUTOENCODER
Autoencoder: unsupervised!

Learning a compact representation of the data (no classification)
First we train an AutoEncoder layer 1.
Second we train an AutoEncoder layer 2.
Autoencoder -> Supervised

Then we train an output layer of non-linearities based on softmax.
Autoencoder -> Supervised

Finally, we fine-tune the whole network in a supervised way.
Denoising stacked Autoencoder: unsupervised

Result = a new latent representation

USDA Equations:
\[
\tilde{x} = qD(x) = \text{Stochastic Dropout} \quad y = f_\theta(\tilde{x}) = s(Wx + b) \quad z = g_\theta'(y) = s(W'y + b')
\]
Denoising stacked Autoencoder: example

**Stage 1.**
**Data-mining stage & Feature extraction:**
Driving Electronic Health Records to build a binary phenotype representation.

**Stage 2.**
**Unsupervised stage:**
Mapping the Binary Patient Representation to get a new space call Deep Patient (or Latent Representation) Using Stacked Denoising Autoencoders.

**Stage 3.**
**Supervised stage:**
Labeling Medical Target and training the Latent Representation by Machine Learning algorithms for classification and prediction of patient's disease.
Supervised Image Segmentation Task

[convolution network diagram with max pooling and deconvolution network diagram]

[Before and After images with text: 'Credit: Facebook']

MS COCO Detection Challenge!

 Credits Matthieu Cord
Partly from COLAH’s Blog http://colah.github.io/posts/2015-08-Understanding-LSTMs/

RECURRENT NEURAL NETWORK
Recurrent Neural Networks have loops.
In the last few years, there have been incredible success applying RNNs to a variety of problems: speech recognition, language modeling, translation, image captioning...
The Problem of Long-Term Dependencies

If we are trying to predict the last word in “the clouds are in the (sky),” we don’t need any further context – it’s pretty obvious the next word is going to be sky.

Consider trying to predict the last word in the text “I grew up in France... I speak fluent (French).” Recent information suggests that the next word is probably the name of a language, but if we want to narrow down which language, we need the context of France, from further back.
The repeating module in a standard RNN contains a single layer.

Unfortunately, as that gap grows, RNNs become unable to learn to connect the information (cf. vanishing gradients). The problem was explored in depth by Hochreiter (1991) and Bengio, et al. (1994), who found some pretty fundamental reasons why it might be difficult. Thankfully, LSTMs/GRUs do not have this problem!
The repeating module in an LSTM/GRU contains four interacting layers.
Sequence modeling with RNNs
One to Many - Image captioning

Human captions from the training set

- A cute little dog sitting in a heart drawn on a sandy beach.
- A dog walking next to a little dog on top of a beach.
- A large brown dog next to a small dog looking out a window.

Automatically captioned

- A dog is sitting on the beach next to a dog.

[Xu et al. 2015]
Many to Many Parallel - Char-nn

Proof. Omitted.

Lemma 0.1. Let $\mathcal{C}$ be a set of the construction.
Let $\mathcal{C}$ be a gerbe covering. Let $\mathcal{F}$ be a quasi-coherent sheaves of $\mathcal{O}$-modules. We have to show that
\[ \mathcal{O}_X(\mathcal{C}) \]

\[ \mathcal{O}_X(\mathcal{F}) = \{ \text{morph}_1 \times \mathcal{O}_X(\mathcal{G}, \mathcal{F}) \} \]
where $\mathcal{G}$ defines an isomorphism $\mathcal{F} \to \mathcal{F}$ of $\mathcal{O}$-modules.

Proof. This is an algebraic space with the composition of sheaves $\mathcal{F}$ on $\mathcal{X}_{\text{etale}}$ we have
\[ \mathcal{O}_X(\mathcal{F}) = \{ \text{morph}_1 \times \mathcal{O}_X(\mathcal{G}, \mathcal{F}) \} \]

Lemma 0.2. This is an integer $\mathcal{Z}$ is injective.

Proof. See Spaces, Lemma ??.

Lemma 0.3. Let $\mathcal{S}$ be a scheme. Let $X$ be a scheme and $X$ is an affine open covering. Let $\mathcal{U} \subset \mathcal{X}$ be a canonical and locally of finite type. Let $X$ be a scheme.

Let $X$ be a scheme which is equal to the formal complex.

The following to the construction of the lemma follows.

Let $X$ be a scheme. Let $X$ be a scheme covering. Let
\[ \mathcal{h} : X \to Y' \to Y \to Y' \times_X Y \to X. \]
be a morphism of algebraic spaces over $S$ and $Y$.

Proof. Let $X$ be a nonzero scheme of $X$. Let $X$ be an algebraic space. Let $\mathcal{F}$ be a quasi-coherent sheaf of $\mathcal{O}_X$-modules. The following are equivalent.

1. $\mathcal{F}$ is an algebraic space over $S$.
2. If $X$ is an affine open covering,

Consider a common structure on $X$ and $X$ the functor $\mathcal{O}_X(U)$ which is locally of finite type.
Many to Many - Machine translation
speech2text

- Input: Audio mp3 (natural language)
- Output: "How much would a woodchuck chuck"

[Chan et al. 2015] [Olah et Carter 2016]
Overview

• Context & Vocabulary
• Unsupervised classification
• Explicit supervised classification
• Implicit supervised classification
• Deep Learning
• Reinforcement Learning
Based on “Introduction to Deep Learning”, by Professor Qiang Yang, from The Hong Kong University of Science and Technology
• What’s Reinforcement Learning?

- Agent interacts with an environment and learns by maximizing a scalar reward
- No labels or any other supervision
- Previously suffering from hand-craft states or representation
Policies and Value Functions

• Policy \( \pi \) is a behavior function selecting actions given states (it defines the probability of each possible action regarding the state \( s \))

\[
a = \arg\max_{\pi(s)} \pi(s)
\]

• Value function \( Q^\pi(s,a) \) is expected total reward \( r \) from state \( s \) and action \( a \) under policy \( \pi \)

\[
Q^\pi(s,a) = \mathbb{E} \left[ r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \ldots \mid s, a \right]
\]

“How good is action \( a \) in state \( s \)?”
Approaches To Reinforcement Learning

- **Policy-based RL**
  - Search directly for the optimal policy $\pi^*$
  - Policy achieving maximum future reward

- **Value-based RL**
  - Estimate the optimal value function $Q^*(s,a)$
  - Maximum value achievable for the best policy $\pi^*$

- **Model-based RL**
  - Build a transition model of the environment
  - Plan (e.g. by look-ahead) using model
Bellman Equation

- Value function can be unrolled recursively
  \[ Q^\pi(s, a) = \mathbb{E}[r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \cdots | s, a] \]
  \[ = \mathbb{E}_{s'} \left[ r_t + \gamma \max_{a'} Q^\pi(s', a') | s, a \right] \]

- Optimal value function \( Q^* (s, a) \) can be unrolled recursively
  \[ Q^*(s, a) = \mathbb{E}_{s'} \left[ r_t + \gamma \max_{a'} Q^*(s', a') | s, a \right] \]
  \[ V_\pi(s) = \max_{a'} Q^\pi(s', a') \]

- Value iteration algorithms solve the Bellman equation
  \[ Q_{i+1}(s, a) = \mathbb{E}_{s'} \left[ r_t + \gamma \max_{a'} Q_i(s', a') | s, a \right] \]

This last equation corresponds to how we should update \( Q \) ideally, i.e. knowing the state distribution \( \text{Prob}(s' | s, a) \) and the complexity is not even polynomial in the number of states.
Deep Reinforcement Learning

- Human

- So what’s DEEP RL?

{Raw Observation, Reward} → Environment → {Actions}
Deep Reinforcement Learning

• Represent value function by deep Q-network with weights $w$

\[ Q(s, a, w) = Q^\pi(s, a) \]

• Define objective function by mean-squared error in Q-values

\[
\mathcal{L}(w_i) = \mathbb{E} \left[ r_t + \gamma \max_{a'} Q(s', a', w_{i-1}) - Q(s, a, w_i) \right]
\]

• Leading to the following Q-learning gradient

\[
\frac{\partial \mathcal{L}(w_i)}{\partial w} = \mathbb{E} \left[ \left( r_t + \gamma \max_{a'} Q(s', a', w_{i-1}) - Q(s, a, w_i) \right) \frac{\partial Q(s, a, w_i)}{\partial w} \right]
\]
DQN in Atari

• End-to-end learning of values $Q(s, a)$ from pixels
• Input state $s$ is stack of raw pixels from last 4 frames
• Output is $Q(s, a)$ for 18 joystick/button positions
• Reward is the change in the score for that step

Mnih, Volodymyr, et al. 2015.
DQN in Atari: Human Level Control

Mnih, Volodymyr, et al. 2015.
AlphaGO: Monte Carlo Tree Search

- MCTS: Model look ahead to reduce searching space by predicting opponent’s moves

\[ V_\pi(s) = \max_a Q_\pi(s, a) \]

**AlphaGO: Learning Pipeline**

- Combine SL and RL to learn the search direction in MCTS

- **SL policy Network**
  - Prior search probability or potential

- **Rollout**:
  - Combine with MCTS for quick simulation on leaf node

- **Value Network**:
  - Build the Global feeling on the leaf node situation

---

Learning to Prune: SL Policy Network

- 13-layer CNN
- Input board position $s$
- Output: $p_\sigma (a|s)$, where $a$ is the next move

Extended Data Table 2 | Input features for neural networks

<table>
<thead>
<tr>
<th>Feature</th>
<th># of planes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stone colour</td>
<td>3</td>
<td>Player stone / opponent stone / empty</td>
</tr>
<tr>
<td>Ones</td>
<td>1</td>
<td>A constant plane filled with 1</td>
</tr>
<tr>
<td>Turns since</td>
<td>8</td>
<td>How many turns since a move was played</td>
</tr>
<tr>
<td>Liberties</td>
<td>8</td>
<td>Number of liberties (empty adjacent points)</td>
</tr>
<tr>
<td>Capture size</td>
<td>8</td>
<td>How many opponent stones would be captured</td>
</tr>
<tr>
<td>Self-atari size</td>
<td>8</td>
<td>How many of own stones would be captured</td>
</tr>
<tr>
<td>Liberties after move</td>
<td>8</td>
<td>Number of liberties after this move is played</td>
</tr>
<tr>
<td>Ladder capture</td>
<td>1</td>
<td>Whether a move at this point is a successful ladder capture</td>
</tr>
<tr>
<td>Ladder escape</td>
<td>1</td>
<td>Whether a move at this point is a successful ladder escape</td>
</tr>
<tr>
<td>Sensibleness</td>
<td>1</td>
<td>Whether a move is legal and does not fill its own eyes</td>
</tr>
<tr>
<td>Zeros</td>
<td>1</td>
<td>A constant plane filled with 0</td>
</tr>
<tr>
<td>Player color</td>
<td>1</td>
<td>Whether current player is black</td>
</tr>
</tbody>
</table>

Feature planes used by the policy network (all but last feature) and value network (all features).
Learning to Prune: RL Policy Network

- 1 Million samples are used to train.
- RL-Policy network VS SL-Policy network.
  - RL-Policy alone wins 80% games against SL-Policy.
  - Combined with MCTS, SL-Policy network is better
- Used to derive the Value Network as the ground truth
  - Making enough data for training
Learning to Prune: Value Network

- **Regression:** Similar architecture
  \[ v^p(s) = \mathbb{E}[z_t | s_t = s, a_t ... T \sim p] \]

- **SL Network:** Sampling to generate a unique game.
- **RL Network:** Simulate to get the game’s final result.

- **Train:** 50 million mini-batches of 32 positions (30 million unique games)
AlphaGO: Evaluation

The version solely using the policy network does not perform any search

QUESTIONS?