Outline of the talk:

- Classifier design.
- Mathematical formulation of the risk describing process of learning.
- Upper bound = guaranteed risk.
- VC-dimension calculation.
- Structural risk minimization.
Classifier design (1)

The object of interest is characterized by observable properties \(x \in X\) and its class membership (unobservable, hidden state) \(y \in Y\), where \(X\) is the space of observations and \(Y\) the set of hidden states.

The objective of a classifier design is to find the optimal decision function \(q^*: X \rightarrow Y\).

Bayesian decision theory solves the problem of minimization of the Bayesian risk

\[
R(q) = \sum_{x,y} p_{XY}(x,y) W(y, q(x))
\]

given the following quantities:

- \(p_{XY}(x,y), \forall x \in X, y \in Y\) — the statistical model of the dependence of the observable properties (measurements) on class membership.
- \(W(y, q(x))\) the loss of decision \(q(x)\) if the true class is \(y\).
Classifier design (2)

Constraints or penalties for different errors depend on the application problem formulation.

However, in applications typically:

- None of the class conditional probabilities (likelihoods) are known, e.g., \( p(x|y) \), \( p(y) \), \( \forall x \in X, y \in Y \).

- The designer is only given a training multi-set \( T = \{(x_1, y_1) \ldots (x_L, y_L)\} \), where \( L \) is the length (size) of the training multi-set.

- The desired properties of the classifier \( q(x) \) are assumed.

Note: Non-Bayesian decision theory offers the solution to the problem if \( p(x|y) \), \( \forall x \in X, y \in Y \) are known, but \( p(y) \) are unknown (or do not exist).
Classifier design via parameter estimation

- **Assume** $p(x, y)$ have a particular form, e.g., a mixture of Gaussians, piece-wise constant, etc., with a finite (i.e., small) number of parameters $\Theta_y$.

- **Estimate** the parameters $\Theta_y$ from the training multi-set $T$.

- **Solve** the classifier design problem (i.e., minimize the risk) by substituting the estimated $\hat{p}(x, y)$ for the true (and unknown) probabilities $p(x, y)$.
  - There is no direct relationship between known properties of estimated $\hat{p}(x, y)$ and the properties (typically the risk) of the obtained classifier $q'(x)$.
  - If the true $p(x, y)$ is not of the assumed form then $q'(x)$ may be arbitrarily bad, even if the size of training set $L$ approaches infinity!

- Implementation is often straightforward, especially if parameters $\Theta_y$ for each class are assumed independent.

- Performance on real data can be predicted empirically from performance on training set (divided to training set and validation set, e.g., crossvalidation).
Learning in statistical pattern recognition

- **Choose a class** $Q$ of decision functions (classifiers) $q: X \rightarrow Y$.

- **Find** $q^* \in Q$ by minimizing some criterion function on the training set that approximates the risk $R(q)$ (which cannot be computed).

- **Learning paradigm** is defined by the approximating criterion function:
  1. **Maximizing likelihood.**
     
     *Example: Estimating the probability density.*
  2. **Using a non-random training set.**
     
     *Example: Image analysis.*
  3. **Empirical risk minimization** in which the true risk is approximated by the error rate on the training set.
     
     *Examples: Perceptron, Neural nets (Back-propagation), etc.*
  4. **Structural risk minimization.**
     
     *Example: SVM (Support Vector Machines).*
Overfitting and underfitting

- How rich class $Q$ of classifiers $q(x, \Theta)$ should be used?
- The problem of generalization is a key problem of pattern recognition: a small empirical risk $R_{\text{emp}}$ need not imply a small true expected risk $R$!

underfit  
fit  
overfit
Asymptotic behavior

- For infinite training data, the law of large number assures
  \[ \lim_{L \to \infty} R_{emp}(\Theta) = R(\Theta). \]

- In general, unfortunately, there is no guarantee for a solution based on the expected risk minimization because
  \[ \arg\min_{\Theta} R_{emp}(\Theta) \neq \arg\min_{\Theta} R(\Theta). \]

Performance on training data is often better than on test data (or real performance).
The idea of the guaranteed risk

- **Idea**: add a prior (called also regularizer).
- This regularizer favors a simpler strategy, cf., Occam razor.
- Vapnik-Chervonenkis learning theory introduces a guaranteed risk $J(\Theta)$, $R(\Theta) \leq J(\Theta)$, with the probabilistic confidence $\eta$.
- The upper bound $J(\Theta)$ may be so large (meaning pessimistic) that it can be useless.
The upper bound of a true risk

- The upper bound was derived by Chervonenkis and Vapnik in the 1970s.
- With the confidence $\eta$, $0 \leq \eta \leq 1$,

$$R(\Theta) \leq J(\Theta) = R_{\text{emp}}(\Theta) + \sqrt{\frac{h \left( \log \left( \frac{2L}{h} \right) + 1 \right)}{L} - \log \left( \frac{\eta}{4} \right)}.$$

where $L$ is the length of the training multi-set, $h$ is the VC-dimension of the class of strategies $q(x, \Theta)$.
- Note that the above upper bound is independent of the true $p(x, y)$!!
- It is the worst case upper bound valid for all possible $p(x, y)$.
- **Structural risk minimization** means minimizing the upper bound $J(\Theta)$.
  *(We will return to structural risk minimization after we explain how to compute VC-dimension.)*
Vapnik-Chervonenkis dimension

- It is a number characterizing the decision strategy.
- Abbreviated VC-dimension.
- It is one of the core concepts in Vapnik-Chervonenkis theory of learning.
- In the original 1974 publication, it was called capacity of a class of strategies.
- The VC dimension is a measure of the capacity of a statistical classification algorithm.
VC-dimension, the idea informally

- The VC-dimension (capacity) of a classification strategy tells how complicated it can be.
- An example: the thresholding a high-degree polynomial. If a very high-degree polynomial is used, it can be very wiggly, and can fit a training set exactly (overfit). Such a polynomial has a high capacity and problems with generalization.
- A linear function, e.g., has a low VC-dimension.

\[
\begin{align*}
    f_1(x) &= (x - 1) \\
    f_2(x) &= (x - 1)(x + 2) \\
    f_3(x) &= (x - 2)(x - 1)(x + 2) \\
    f_6(x) &= (x - 2)(x - 1)x(x + 1)(x + 2)(x + 3)
\end{align*}
\]

Light green circles symbolize data points.
Consider a classification strategy \( q \) with some parameter vector \( \Theta \).

The strategy \( q \) can shatter a set of data points \( x_1, x_2, \ldots, x_n \) if, for all possible assignments of labels \( y \in Y \) to data points, there exists a parameter \( \Theta \) such that the model \( q \) makes no errors when evaluating that set of data points.

Shattering example: \( q \) is a line in a 2D feature space.

3 points, shattered

4 points, undivisible
Consider a set of dichotomic strategies $q(x, \Theta) \in Q$.

The set consisting of $h$ data points (observations) can be labelled in $2^h$ possible ways.

A strategy $q \in Q$ exists which assigns labels correctly to all possible configurations.

(Process of finding all possible configurations with correctly assigned labels is called shattering.)

VC-dimension (definition) is the maximal number $h$ of data points (observations) that can be shattered.
VC-dimension of a linear strategy in a 2D feature space

- A set of parameters $\Theta = \{\Theta_0, \Theta_1, \Theta_2\}$.
  A linear strategy $q(x, \Theta) = \Theta_1 x_1 + \Theta_2 x_2 + \Theta_0$.

- Shattering example (revisited):

  3 points, shattered
  4 points, undivisible

- 3 points in 2D space ($n = 2$) can be shattered.
  There was counter example given that 4 points cannot be shattered.
  $\Rightarrow$ VC-dimension $h = 3$. 
VC-dimension for a linear strategy in a $n$-dimensional space

A special case, $n=2$.

VC-dimension $= 3$.

Generalization to $n$-dimensions for linear classifiers

- A hyperplane in the space $\mathbb{R}^n$ shatters any set of $h = n + 1$ linearly independent points.

- Consequently, VC-dimension of linear decision strategies is $h = n + 1$. 
VC-dimension in a 2D space for a circular strategy

Maximally 4 data points in $\mathbb{R}^2$ can be shattered in 8 possible ways $\Rightarrow$ VC-dimension $h = 4$. 
Small # of parameters, VC-dimension=∞

Counterexample by E. Levin, J.S. Denker (Vapnik 1995):

- A sinusoidal 1D classifier, \( q(x, \Theta) = \text{sign}(\sin(\Theta x)) \), \( x, \Theta \in \mathbb{R} \).
- For any given number \( L \in \mathbb{N} \), the points \( x_i = 10^{-i}, i = 1, \ldots, L \) and be found and arbitrary labels \( y_i, y_i \in \{-1, 1\} \) can assigned to \( x_i \).
- Then \( q(x, \Theta) \) is the correct labelling if \( \Theta = \pi \left( 1 + \sum_{i=1}^{L} \frac{(1-y_i)10^i}{2} \right) \).

Example: \( L = 3, y_1 = -1, y_2 = 1, y_3 = -1 \).

- Thus the VC dimension of this decision strategy is infinite.
Examples of other VC-dimension $= \infty$ strategies

- Nearest-neighbor classifier – any number of observations, labeled arbitrarily, will be classified. Thus VC-dimension $= \infty$. Also $R_{emp} = 0$. The VC-dimension provides no information in this particular case.

- Convex polygons classifying observation lying on a circle, VC-dimension $= \infty$.

- SVM classifiers with Gaussian (or RBF ...) kernel, VC-dimension $= \infty$. 
Structural risk minimization

- Minimize guaranteed risk $J(\Theta)$, that is the upper bound

$$R(\Theta) \leq J(\Theta) = R_{\text{emp}}(\Theta) + \sqrt{\frac{h \left( \log \left( \frac{2L}{h} \right) + 1 \right) - \log \left( \frac{n}{4} \right)}{L}}.$$ 

For each model $i$ in the list of hypotheses

- Compute its VC-dimension $h_i$.
- $\Theta_i^* = \arg\min_{\Theta_i} R_{\text{emp}}(\Theta_i)$.
- Compute $J_i(\Theta_i^*, h_i)$.

Choose the model with the lowest $J_i(\Theta_i^*, h_i)$.

- Preferably, optimize directly over both $(\Theta^*, h^*) = \arg\min_{\Theta, h} J(\Theta, h)$.

- Gap tolerant linear classifiers minimize $R_{\text{emp}}(\Theta)$ while maximizing margin. Support Vector Machine does just that.
Structural risk minimization pictorially

- Guaranteed risk $J(\Theta, h)$
- Empirical risk
- Regularizer
- Risks

VC-dimension $h$

Space of nested hypotheses with decreasing $h$
VC-dimension, a practical view

**Bad news:** Computing the guaranteed risk is useless in many practical situations.

- VC dimension cannot be accurately estimated for non-linear models such as neural networks.
- Structural Risk Minimization may lead to a non-linear optimization problem.
- VC dimension may be infinite (e.g., for a nearest neighbor classifier), requiring infinite amount of training data.

**Good news:** Structural Risk Minimization can be applied for linear classifiers.

- Especially useful for Support Vector Machines.
Empirical risk minimization, notes

Is then empirical risk minimization = minimization of training set error, e.g., neural networks with backpropagation, dead? **No!**

- Guaranteed risk $J$ may be so large that this upper bound becomes useless.
  
  *Find a tighter bound and you will be famous! It is not impossible!*

- Vapnik, Chervonenkis suggest learning with progressively more complex classes of the decision strategies $Q$.

- Vapnik & Chervonenkis’ theory justifies using empirical risk minimization on classes of functions with a reasonable VC dimension.

- Empirical risk minimization is computationally hard (impossible for large $L$). Most classes of decision functions $Q$ for which the empirical risk minimization (at least locally) can be efficiently organized are often useful.
  
  *Where does the nearest neighbor classifier fit in the picture?*