Lecture 1: Basic Concepts of Machine Leaning

Cognitive Systems II - Machine Learning

Ute Schmid (lecture)

Emanuel Kitzelmann (practice)

(Based on slides prepared March 2005 by Maximilian Röglinger)

Applied Computer Science, Bamberg University

last change October 16, 2007

Organization of the Course

- **Homepage**:
  `http://www.cogsys.wiai.uni-bamberg.de/teaching/`
- **Textbook**: Tom Mitchell (1997). Machine Learning. McGraw Hill.
- Practice
  - Programming Assignments in Java and Prolog
  - **Marked exercise sheets and extra points for the exam**

Outline of the Course

- Basic Concepts of Machine Learning
- Basic Approaches to Classification Learning
  - Foundations of Classification Learning
  - Decision Trees
  - Perceptrons and Multilayer-Perceptrons
  - Human Concept Learning
- Special Aspects of Classification Learning
  - Inductive Logic Programming
  - Genetic Algorithms
  - Instance-based Learning
  - Bayesian Learning
  - Kernel Methods

Outline of the Course

- Theoretical Aspects of Learning
  - Evaluating Hypotheses
  - Computational Learning Theory
- Learning Programs and Strategies
  - Reinforcement Learning
  - Inductive Function Synthesis
  - Analytical Learning
- Further Topics and Applications in Machine Learning (e.g. data mining)

Course Objectives

- Introduce central approaches of machine learning
- Point out relations to human learning
- Provide understanding of the fundamental structure of learning problems and processes
- **Explore algorithms that solve such problems**

Some Quotes as Motivation

If an expert system–brilliantly designed, engineered and implemented–cannot learn not to repeat its mistakes, it is not as intelligent as a worm or a sea anemone or a kitten.

Oliver G. Selfridge, from The Gardens of Learning

If we are ever to make claims of creating an artificial intelligence, we must address issues in natural language, automated reasoning, and machine learning.

George F. Luger

What is Machine Learning?

- Some definitions
  - *Machine learning refers to a system capable of the **autonomous acquisition** and integration of knowledge. This capacity to **learn from experience, analytical observation**, and other means, results in a system that can **continuously self-improve** and thereby offer **increased efficiency and effectiveness**.*

    `http://www.aaai.org/AITopics/html/machine.html`

  - *The field of machine learning is concerned with the question of how to **construct computer programms** that **automatically improve** with experience.*

    Tom M. Mitchell, Machine Learning (1997)

ML as Multidisciplinary Field

Machine learning is inherently a **multidisciplinary** field

- artificial intelligence
- probability theory, statistics
- computational complexity theory
- information theory
- philosophy
- psychology
- neurobiology
- ...

e.g. CALD (Center of Automated Learning and Discovery at CMU)

Knowledge-based vs. Learning Systems

Knowledge-based Systems: Acquisition and modeling of common-sense knowledge and expert knowledge

$\Rightarrow$ limited to given knowledge base and rule set

$\Rightarrow$ Inference: **Deduction** generates no new knowledge but makes implicitly given knowledge explicit

$\Rightarrow$ **Top-Down**: from rules to facts

Learning Systems: Extraction of knowledge and rules from examples/experience

- Teach the system vs. program the system
- Learning as **inductive process**

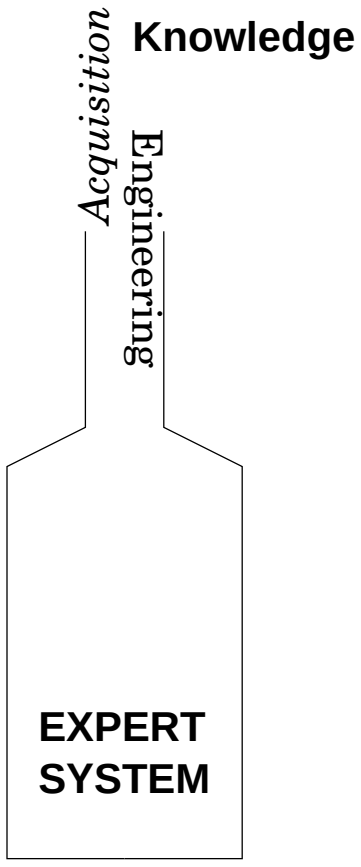
$\Rightarrow$ **Bottom-Up**: from facts to rules

Knowledge-based vs. Learning Systems

$\Rightarrow$ *A flexible and adaptive organism cannot rely on a fixed set of behavior rules but must learn (over its complete life-span)!*

$\Rightarrow$ Motivation for Learning Systems

Knowledge Acquisition Bottleneck



(Feigenbaum, 1983)

- Break-through in computer chess with *Deep Blue*: Evaluation function of chess grandmaster Joel Benjamin.
  Deep Blue cannot change the evaluation function by itself!
- Experts are often not able to verbalize their special knowledge.
  $\Rightarrow$ Indirect methods: Extraction of knowledge from expert *behavior* in *example* situations
  (diagnosis of X-rays, controlling a chemical plant, ...)

Learning as Induction

Deduction		Induction	
All humans are mortal.	(Axiom)	Socrates is human.	(Background K.)
Socrates is human.	(Fact)	Socrates is mortal.	(Observation(s))
Conclusion:		*Generalization:*	
Socrates is mortal.		All humans are mortal.	

Deduction: from general to specific $\Rightarrow$ **proven** correctness

Induction: from specific to general $\Rightarrow$ (**unproven**) knowledge gain

Induction generates hypotheses
not knowledge!

Epistemological problems

$\Rightarrow$ pragmatic solutions

- *Confirmation Theory:* A hypothesis obtained by generalization gets supported by new observations (not proven!).
- *grue* Paradox:
  All emeralds are grue.
  Something is grue, if it is green before a future time t and blue thereafter.
  $\Rightarrow$ Not learnable from examples!

Inductive Learning Hypothesis

- As shown above inductive learning is **not** proven correct
- The learning task is to determine a hypothesis $h \in H$ identical to the target concept c for all possible instances in instance space X
  $(\forall x \in X)[h(x) = c(x)]$
- Only training examples $D \subset X$ are available
- Inductive algorithms can at best guarantee that the output hypothesis h fits the target concept over D
  $(\forall x \in D)[h(x) = c(x)]$

$\Rightarrow$ **Inductive Learning Hypothesis**: Any hypothesis found to approximate the target concept well over a sufficiently large set of training examples will also approximate the target function well over other unobserved examples

Merit of Machine Learning

- Great practical value in many application domains
- Data Mining: large databases may contain valuable implicit regularities that can be discovered automatically (outcomes of medical treatments, consumer preferences)
- Poorly understoon domains where humans might not have the knowledge needed to develop efficient algorithms (human face recognition from images)
- Domains where the program must dynamically adapt to changing conditions (controlling manufacturing processes under changing supply stocks)

Concept and Classification Learning

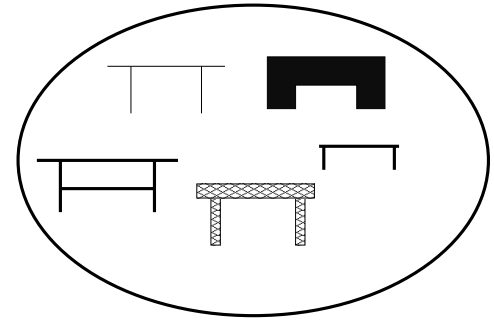
Concept learning:

- Objects are clustered in concepts.

  Extensional: (infinite) set X of all exemplars

  Intensional: finite characterization

  T $= \{x |$ has-3/4-legs(x), has-top(x)$\}$



- Construction of a finite characterization from a subset of examples in X ("training set" D).
- $h : X \rightarrow \{0, 1\}$

Classification learning:

- Identification of relevant attributes and their interrelation, which characterize an object as member of a class.
- $h : X \rightarrow K$

Constituents of Classification Learning

- A set of training examples $D \subset X$
  Each example is represented by an n-ary feature vector $x \in X$
  and associated with a class $c(x) \in K$: $\langle x, c(x) \rangle$
- A learning algorithm constructing a hypothesis $h \in H$
- A set of new objects, also represented by feature vectors which can be classified according to h

Examples for features and values

- Sky $\in$ {sunny, rainy}
- AirTemp $\in$ {warm, cold}
- Humidity $\in$ {normal, high}

Concept Learning / Examples

- Occurence of Tse-Tse fly yes/no, given geographic and climatic attributes
- Risk of cardiac arrest yes/no, given medical data
- Credit-worthiness of customer yes/no, given personal and customer data
- Safe chemical process yes/no, given physical and chemical measurements

- Generalization of pre-classified example data, application for prognosis

Learning Terminology

- Supervised learning: pre-classified examples
- Unsupervised learning: no classification available (data exploration)

Different approaches

- Concept/Classification vs. Policy Learning
- Symbolic vs. Statistical/Neural Network Learning
- Inductive vs. Analytical Learning

Some General Learning Strategies

- rote learning/learning by being told (no generalization/induction)
- learning by analogy (generalization over base and target problem)
- learning from discovery (unsupervised learning)
- learning from experience
- learning from examples (classical inductive approach)

Further Example Learning Problems

- Handwriting recognition
- Play checkers
- Robot driving

Designing a Learning System

- **Learning system**: A computer program is said to **learn** from *experience* E with respect to some *class of tasks* T and *performance measure* P, if its performance at tasks in T, as measured by P, improves with experience E.

- i.e. Handwriting recognition
  - T: recognizing and classifying handwritten words within images
  - P: percent of words correctly classified
  - E: database of handwritten words with given classifications

Designing a Learning System

- consider designing a program to learn to recognize handwritten words in order to illustrate some of the basic design issues and approaches to machine learning

1. **Choosing the Training Experience**
   - direct or indirect feedback
   - degree to which the learner controls the sequence of training examples
   - representativity of the distribution of the training examples

   $\Rightarrow$ significant impact on success or failure

Designing a Learning System

2. **Choosing the Target Function**
   - determine what type of knowledge will be learned
   - most obvious form is some kind of combination of feature values which can be associated with a class (word/letter)
3. **Choosing a Representation for the Target Function**
   - e.g. a large table, a set of rules, a linear function, an arbitrary function
4. **Choosing a Learning Algorithm**
   - Decision Tree, Multi-Layer Perceptron, ...
5. **Presenting Training Examples**
   - all at once
   - incrementally

Recapitulation: Notation

- **Instance Space** X: set of all possible examples over which the concept is defined (possibly attribute vectors)
- **Target Concept** $c : X \rightarrow \{0, 1\}$: concept or function to be learned
- **Training example** $x \in X$ of the form $< x, c(x) >$
- **Training Set** D: set of all available training examples
- **Hypothesis Space** H: set of all possible hypotheses according to the hypothesis language
- **Hypothesis** $h \in H$: boolean valued function of the form $X \rightarrow \{0, 1\}$ or $X \rightarrow K$

$\Rightarrow$ the goal is to find a $h \in H$, such that $(\forall x \in X)[h(x) = c(x)]$

Hypothesis Language

- H is determined by the predefined language in which hypotheses can be formulated
- e.g.: Conjunctions of feature values
  vs. Disjunction of conjunctions
  vs. matrix of real numbers
  vs. Horn clauses
  ...
- Hypothesis language and learning algorithm are highly interdependent
- Each hypothesis language implies a bias!

Properties of Hypotheses

- general-to-specific ordering
  - naturally occuring order over H
  - learning algorithms can be designed to search H exhaustively without explicitly enumerating each hypothesis h
  - h_i is **more_general_or_equal_to** h_k (written $h_i \geq_g h_k$)
    $\Leftrightarrow (\forall x \in X)[(h_k(x) = 1) \rightarrow (h_i(x) = 1)]$
  - h_i is (stricly) **more_general_to** h_k (written $h_i >_g h_k$)
    $\Leftrightarrow (h_i \geq_g h_k) \wedge (h_k \not\geq_g h_i)$
  - $\geq_g$ defines a **partial ordering** over the Hypothesis Space H

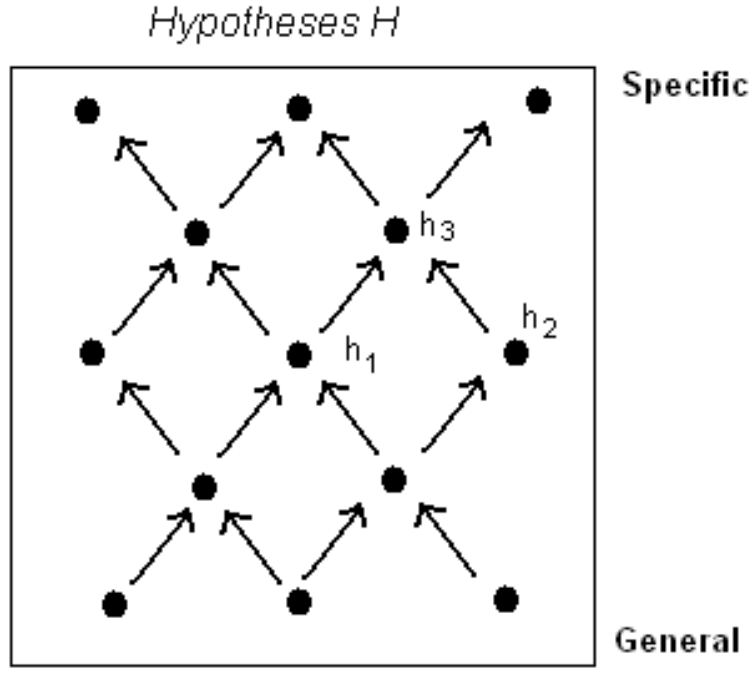
Running Example

- example target concept $Enjoy$: *"days on which Aldo enjoys his favorite sport"*
- set of example days D, each represented by a set of attributes

Example	Sky	$AirTemp$	$Humidity$	$Wind$	$Water$	$Forecast$	$Enjoy$
1	Sunny	Warm	Normal	Strong	Warm	Same	Yes
2	Sunny	Warm	High	Strong	Warm	Same	Yes
3	Rainy	Cold	High	Strong	Warm	Change	No
4	Sunny	Warm	High	Strong	Cool	Change	Yes

- the task is to learn to predict the value of $Enjoy$ for an arbitrary day, based on the values of its other attributes

Properties of Hypotheses - Example



$h_1 =$ Aldo loves playing Tennis if the sky is sunny

$h_2 =$ Aldo loves playing Tennis if the water is warm

$h_3 =$ Aldo loves playing Tennis if the sky is sunny and the water is warm

$\Rightarrow h_1 >_g h_3, h_2 >_g h_3, h_2 \not>_g h_1, h_1 \not>_g h_2$

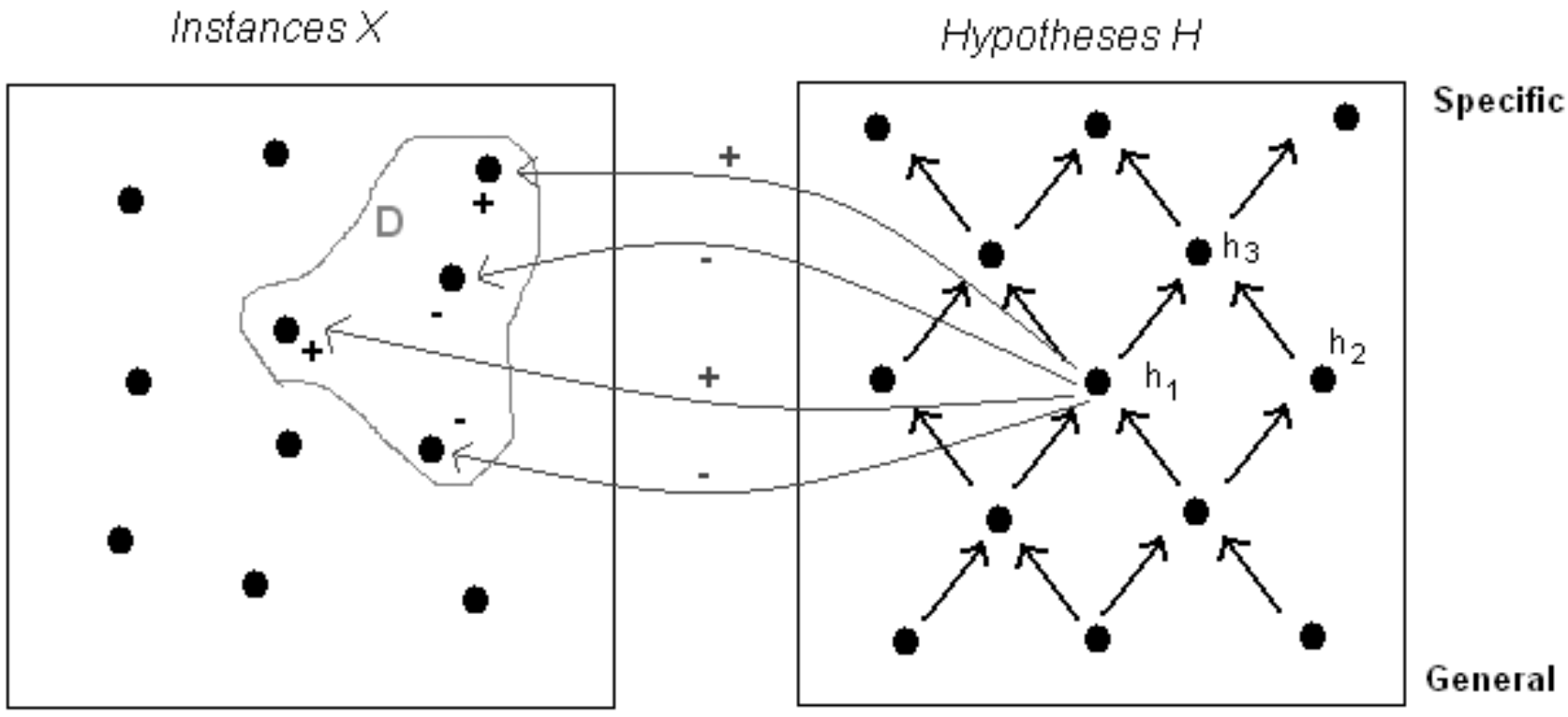
Properties of Hypotheses

- consistency
  - a hypothesis h is **consistent** with a set of training examples D iff $h(x) = c(x)$ for each example $< x, c(x) >$ in D

$$Consistent(h, D) \equiv (\forall < x, c(x) > \in D)[h(x) = c(x)]$$

  - that is, every example in D is classified correctly by the hypothesis

Properties of Hypotheses - Example



h_1 is consistent with D

Learning Involves Search

- Searching through a space of possible hypotheses to find the hypothesis that best fits the available training examples and other prior constraints or knowledge
- Different learning methods search different hypothesis spaces
- Learning methods can be characterized by the conditions under which these search methods converge toward an “optimal” hypothesis