Lecture 1: Basic Concepts of Machine Leaning

Cognitive Systems II - Machine Learning

SS 2005

Ute Schmid (lecture)

Emanuel Kitzelmann (practice)

Maximilian Roeglinger (tutor)

Applied Computer Science, Bamberg University

Organization of the Course

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

Outline of the Course

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

Outline of the Course

- Learning Programs and Strategies
  - Reinforcement Learning
  - Inductive Function Synthesis
- Further Topics and Applications in Machine Learning

Course Objectives

- introduce to the central approaches of machine learning
- point out relations to human learning
- define a class of problems that encompasses interesting forms of learning
- explore algorithms that solve such problems
- provide understanding of the fundamental structure of learning problems and processes

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

Find a bug in a program, and fix it, and the program will work today. Show the program how to find and fix a bug, and the program will work forever.

Oliver G. Selfridge, in AI's Greatest Trends and Controversies

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 in concerned with the question of how to* ***construct computer programms*** *that* ***automatically improve*** *with experience.*

    Tom M. Mitchell, Machine Learning (1997)

What is Machine Learning?

- machine learning is inherently a **multidisciplinary** field
  - i.e. artificial intelligence, probability, statistics, computational complexity theory, information theory, philosophy, psychology, neurobiology, ...
- Knowledge-based vs. Learning Systems
- Different approaches
  - Concept vs. Classification Learning
  - Symbolic vs. Statistical Learning
  - Inductive vs. Analytical Learning

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

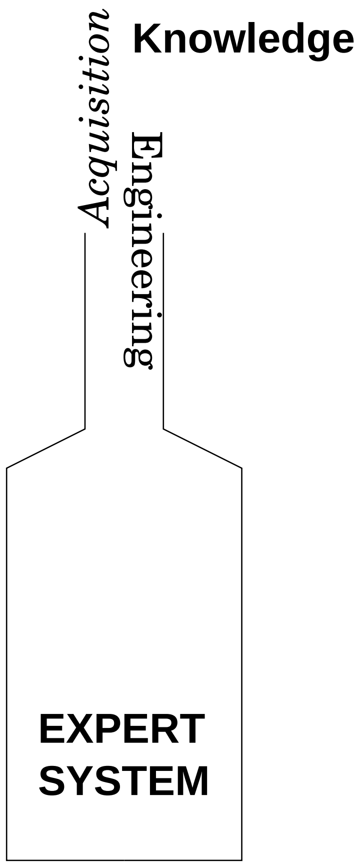
- Teach the system vs. program the system
- Learning as **inductive process**
- $\Rightarrow$ **Bottom-Up**: from facts to rules

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
- nevertheless the learning task is determine a hypothesis $h \in H$ identical to the target concept c
  $(\forall x \in X)[h(x) = c(x)]$
- only training examples D are available
- inductive algorithms can at best guarantee that the output hypothesis D fits the target concept over D
  $(\forall x \in D)[h(x) = c(x)]$

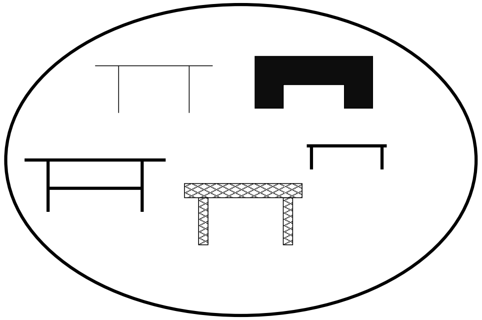
$\Rightarrow$ **Inductive Learning Hypothesis**: Any hypothesis found to approximate the target concept well over a sufficiently large set of training examples will also approximate the target function well over other unobserved examples

Concept vs. Classification Learning

Concept learning:

- Objects are clustered in concepts.

  Extensional: (infinite) set 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 ("training set").

Classification learning:

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

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-worthinessof 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

Symbolic vs. Statistical Learning

Symbolic Learning: underlying learning strategies such as rote learning, learning by being told, learning by analogy, learning from examples and learning from discovery

- knowledge is represented in the form of symbolic descriptions of the learned concepts, i.e. production rules or concept hierarchies
- i.e. decision trees, inductive logic programming

Statistical Learning: modeling of the behaviour

- sometimes called statistical inference, connectionstic learning
- i.e. Artificial Neuronal Networks, Bayesian Learning

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. a checkers learning problem
  - T: playing checkers
  - P: percent of games won against opponents
  - E: playing practice games against itself

Designing a Learning System

- consider designing a program to learn to play checkers 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 a function, that chooses the best move for any given board state
- i.e., $ChooseMove : B \longrightarrow M$

- sometimes evaluation functions are easier to learn
- i.e., $V : B \longrightarrow \Re$
- assigns higher values to better boards states by recursively generating the successor states and evaluating them until the game is over

- no efficient solution (nonoperational definition)

$\Rightarrow$ **function approximation is sufficient (operational definition)**

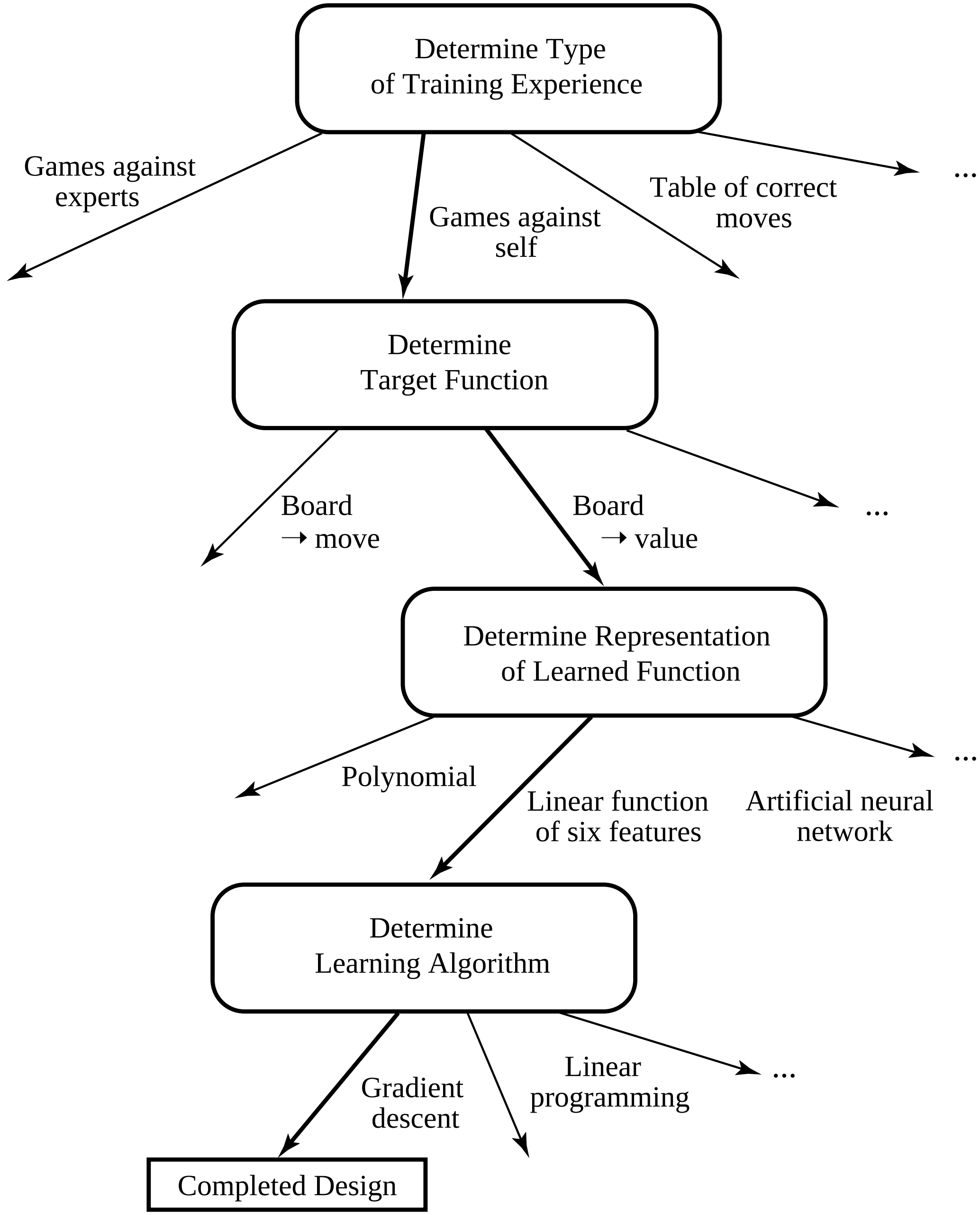
Designing a Learning System

3. **Choosing a Representation for the Target Function**
    - e.g. a large table, a set of rules or a linear function
    - e.g. $V(b) = w_0 + w_1 x_1 + w_2 x_2 + ... + w_i x_i$

4. **Choosing a Function Approximation Algorithm**

5. **Estimating Training Values**
    - it is easy to assign a value to a final state
    - the evaluation of *intermediate* states is more difficzult

Designing a Learning System



Notation

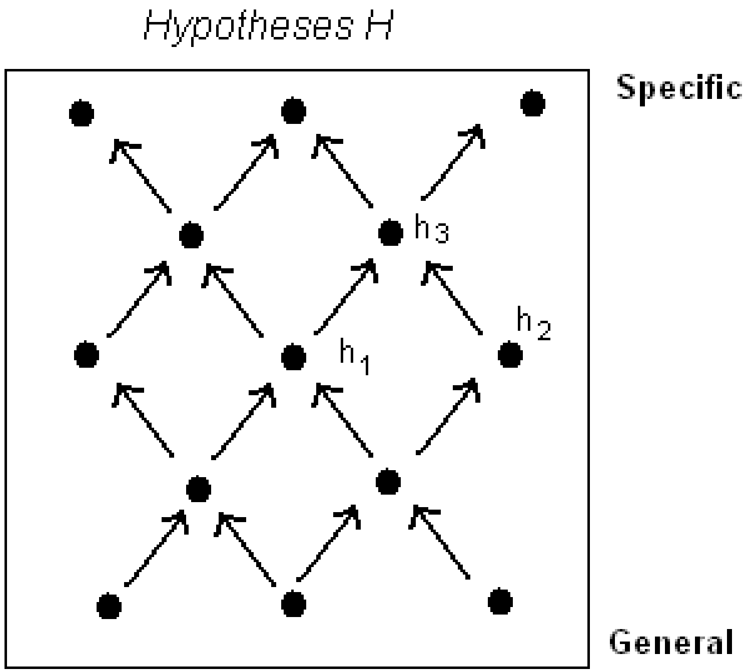
- **Instance Space** X: set of all possible examples over which the concept is defined (possibly attribute vectors)
- **Target Concept** $c : x \rightarrow \{0, 1\}$: concept or function to be learned
- **Training example** $x \in X$ of the form $< x, c(x) >$
- **Training Set** D: set of all available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

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

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 $h_i \geq_g h_k$)
    $\Leftrightarrow (\forall x \in X)[(h_k(x) = 1) \rightarrow (h_j(x) = 1)]$
  - h_i is (stricly) **more_general_to** h_k (written $h_i >_G h_k$)
    $\Leftrightarrow (h_j \geq_g h_k) \wedge (h_k \not\geq_g h_i)$
  - $\geq_g$ defines a **partial ordering** over the Hypothesis Space H

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

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

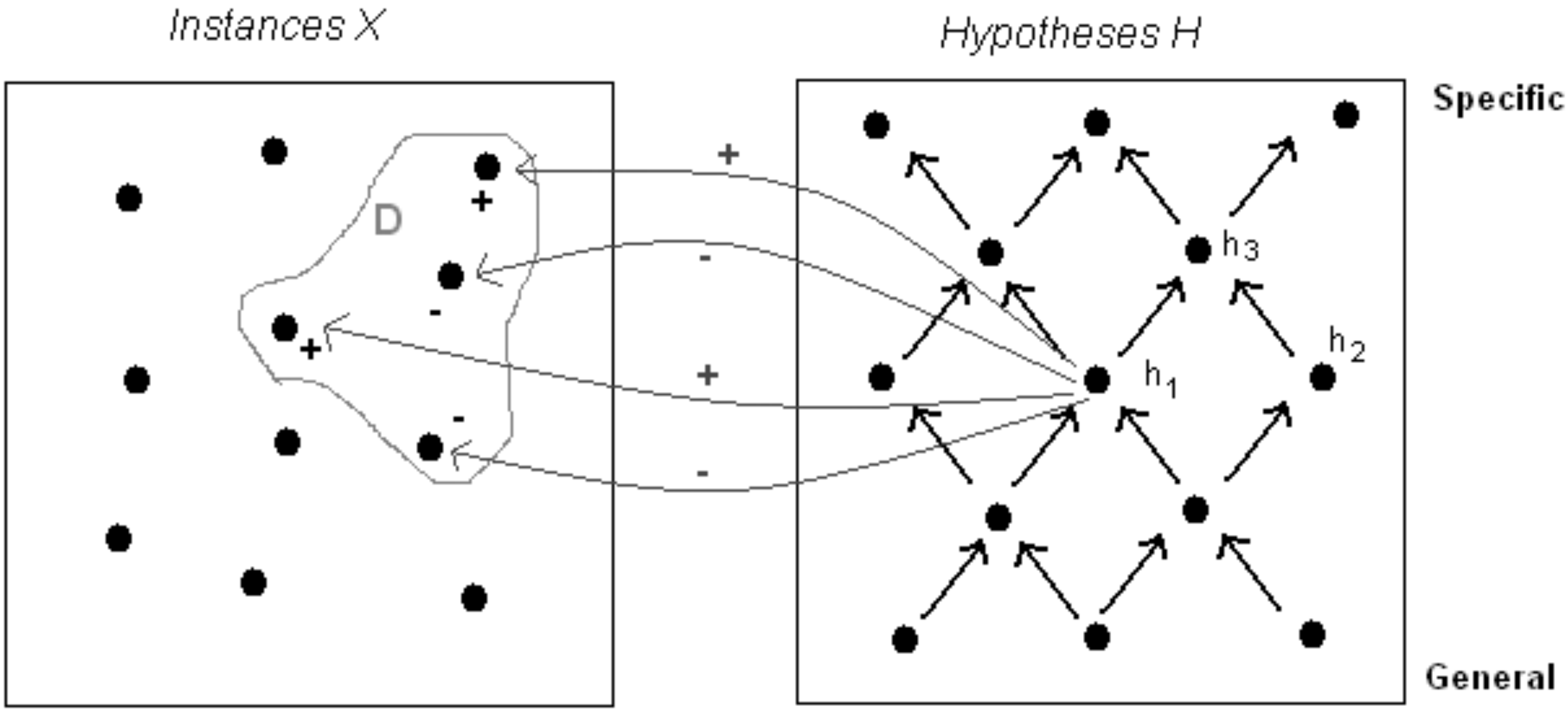
Properties of Hypotheses

- 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