Knowledge Acquisition and Machine Learning

Katharina Morik

two meanings of knowledge acquisition
underlying assumptions of knowledge acquisition and machine learning -
some terminological confusion
consequences of the assumptions
(and the confusion)
some experience with integration

Two Meanings

"Machine learning is knowledge acquisition."
By learning a machine acquires knowledge.

"Machine learning needs knowledge acquisition."
Before starting the machine to learn, the user has to do something where knowledge acquisition techiques could be of help.

What do knowledge acquisition techniques offer?
Where do they point at?
What is the situation underlying both statements?

Underlying Assumptions

concerning:

knowledge-based systems
knowledge, skill, expertise
model, modeling, knowledge acquisition

consequences for:

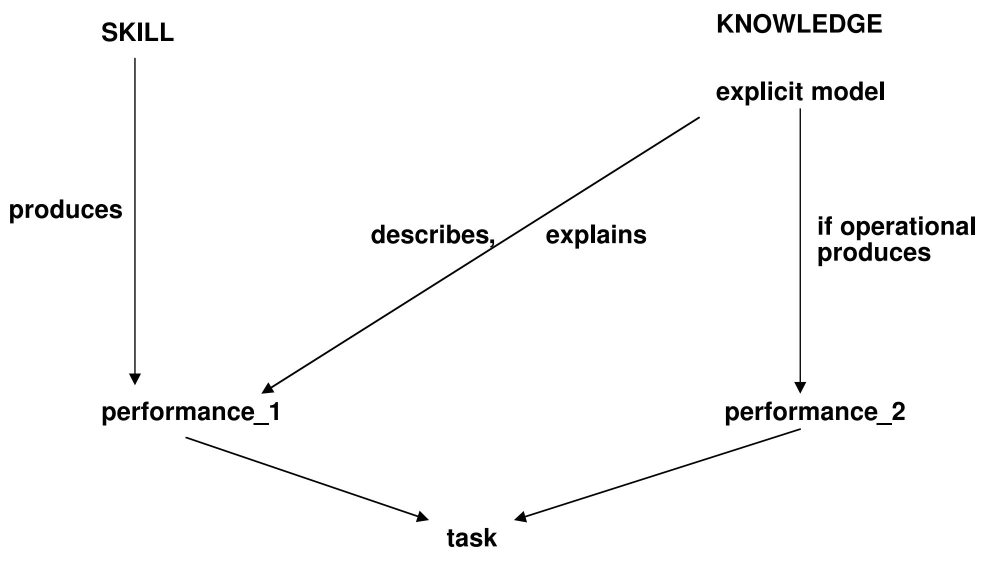
the roles of expert, knowledge engineer, acquisition system, performance system
(desired) functionality of the acquisition system

Knowledge-Based Systems

Computer systems can provide for an inference machine which interprets a knowledge base.
The more knowledge is in the knowledge base the better performs the system.
The knowledge in the knowledge base is understandable and interpretable by human non computer scientists.

Goals:
transparency of computer systems: explicit and understandable
separation of stored items and control
interpretation of the items by the machine (not only by the user)
representing cognitive theories by computer programs

Overall Framework



Knowledge

transfer view:
basis of expertise, expert's problem solving knowledge

performance view:
those items in a computer system that enhance its performance

knowledge level view:
knowledge is whatever can be ascribed to an agent such that its behavior can be computed according to the principle of rationality

Skill as Basis of Expertise

If expertise is based on skill instead of knowledge,
- this would explain the observations that
  experts report incompletely, inconsistent with what can be observed as their behavior, and always change their mind
- because then
  interviewing experts <u>starts</u> their activity of constructing knowledge and, thus, gives us naive knowledge.

"The danger for knowledge acquisition is that we build systems with naive knowledge bases and confuse this with a knowledge acquisition problem!"

(Morik 1989)

Why call it "knowledge"?

If we invert the principle "more knowledge - better performance" to
"whatever enhances performance" must be knowledge
we subsume good programming and good hardware under "knowledge".

Why not call good guidelines and tools for software engineering at a high level what they are?
Why not call the system that accepts user input the input device?

"By defining knowledge in terms of problem-solving performance, it was possible to convert 'acquisition of knowledge' into 'improvement of performance'."
(Dietterich 1986)

Paradox of Ascription

Ascribing knowledge to an agent means to state:
"the agent in fact has this knowledge."
Stating that it is an ascription means the observer creates this knowledge.

Claiming an isomorphic relation between description and described helps - but can we believe it?

Way out:
Knowledge is a consistent description of observed phenomena which allows to explain future observations.
Operational knowledge generates what can be compared with observations or has a value in itself.

Constructive View of Knowledge

- knowledge is explicit and explicable
- knowledge is not the same as skill but may describe skill
- knowledge is generated by a community of observers
- knowledge describes and explains observations
- knowledge is as much determined by the observations as by the observers
- the observers use their (background) knowledge in order to
  select relevant observations and
  create new knowledge
- knowledge changes in time
- knowledge is validated
  by comparison with observations and
  by showing its use and usefulness.

CYCLE!

Model - Modeling

transfer view:
the model is already there - only needs to be translated into another form
there is no modeling

performance view:
the model is a guideline, a generic structure, a frame which is made by the system developers and filled by the knowledge engineers
modeling is a software engineering task outside the scope of the system

constructive view:
the model is a set of knowledge items, theory
to be constructed interactively by expert, knowledge engineer, system
modeling is theory formation, i.e.
forming and revising interactively a model for a set of observations
including setting up this set of observations (aggregation)
because of its cyclic nature modeling cannot use stepwise refinement but needs full reversibility

Knowledge Acquisition

transfer view:
extracting knowledge from human experts, transferring the knowledge from one representation into another, operational one

performance view:
building a knowledge-based system
using prepared generic structures to specify system behavior, thus reducing knowledge acquisition to filling in a generic structure

constructive view:
interactive and reversible modeling (theory formation)

Roles of XP, KE, Systems in Manual Knowledge Acquisition

transfer view:
- expert and performance system are both containers of knowledge,
- the knowledge engineer with the support of the acquisition system *ferrits* information out of the expert and represent it in an appropriate formalism
- more or less replacing the knowledge engineer by the acquisition system
- acquisition system plays an initiative, querying role

performance view:
- computer scientist develops formalized generic structures maintained by a system
- the expert fills in the structures with the help of the acquisition system
- possibly, the performance system makes good use of the generic structures
- acquisition system is a structure-oriented editor

Roles of XP, KE, and Systems in Automatic Knowledge Acquisition

classical learning from examples:
- knowledge engineer agrees with the expert on example set, orders the examples and designs the appropriate representation
- expert classifies examples
- learning system (re-)constructs a model (concepts) which explains the expert's classification

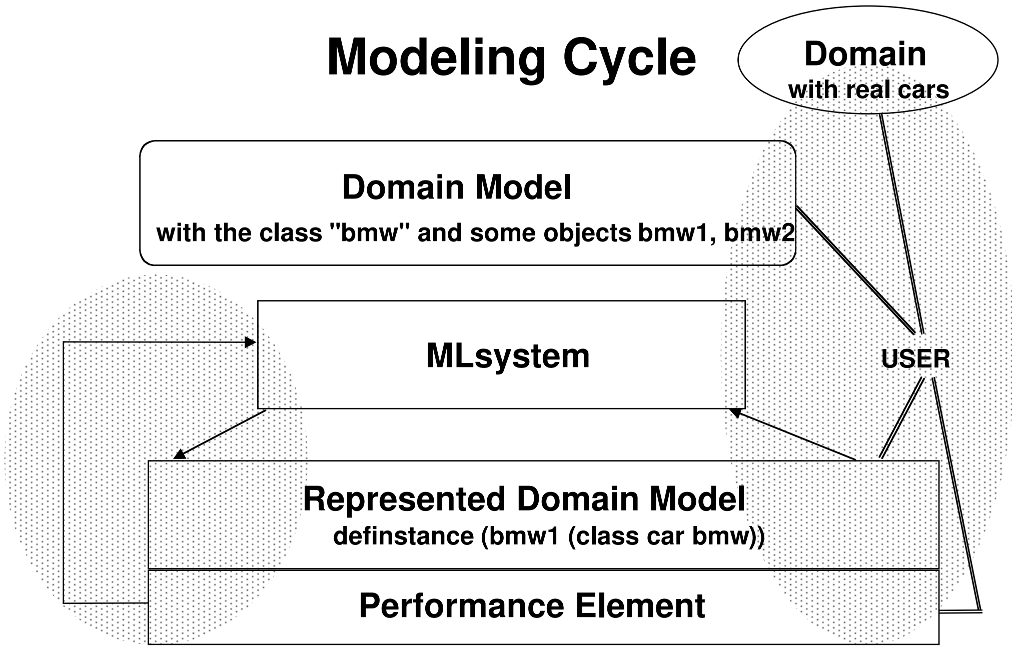
performance enhancing learning:
- performance system solves problems and interacts with an expert
- expert gives solution
- learning system (re-)constructs the solution in the formalism of the performance system

... in the Constructive View

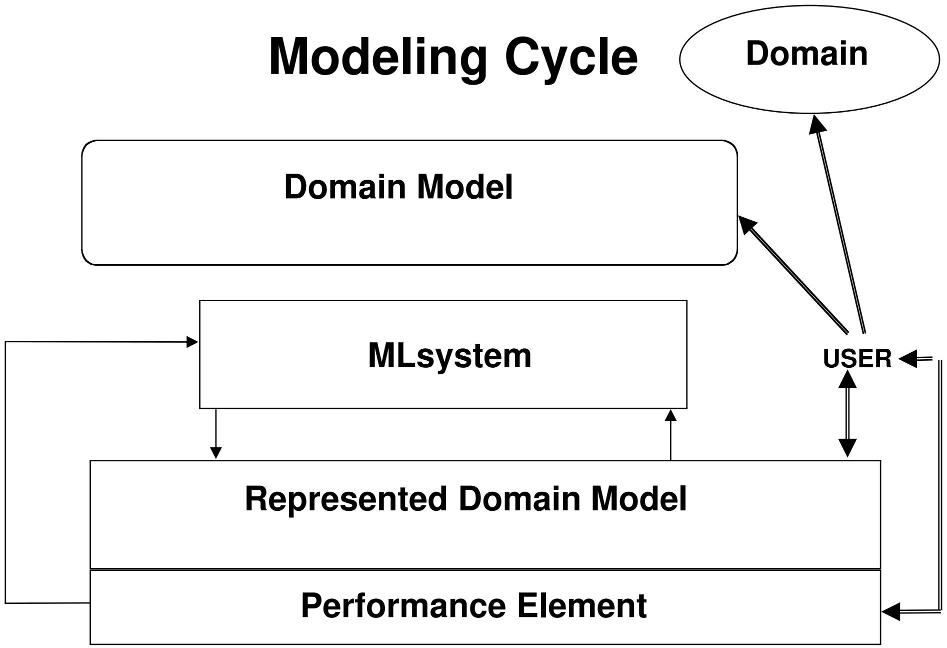
manual acquisition:
- expert and knowledge engineer construct the model, they are initiative
- acquisition system offers
  - powerful representation formalism
  - consistency and integrity checks, knowledge revision
  - book-keeping of the modeling with propagation of consequences

automatic acquisition:
- knowledge engineer prepares the representation language
- expert provides observations
- learning system
  - aggregates observations
  - forms new concepts and builds a structure relating them
  - introduces new terms into the representation language
  - constructs a model which is consistent with the observations

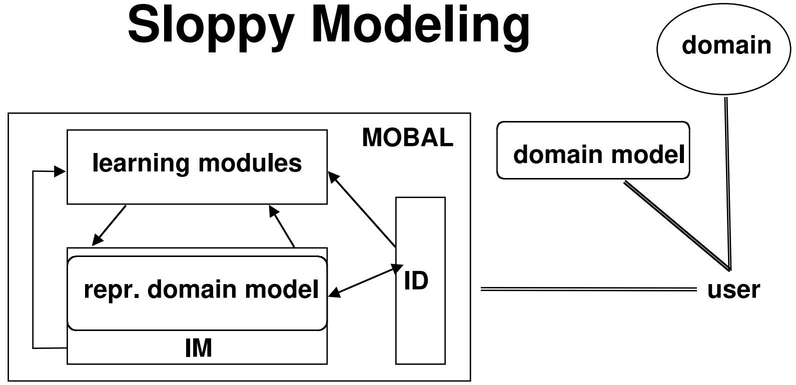
Modeling Cycle



Modeling Cycle



Sloppy Modeling



The input may be false	system handles noisy data
the evolved domain model may entail contradictions	system checks consistency and revises knowledge
descriptors may be missing	system introduces new terms
background knowledge may be missing	system also learns background knowledge

The Art of Machine Learning

There is a lot of preparation work:
- select the well-suited descriptors,
- encode them appropriately,
- select the well-suited examples,
- design the background knowledge.

All things may turn out to be wrong or become wrong:
- recognize that there is a descriptor missing,
- introduce a new descriptor,
- check consistency of the represented domain model, revise the knowledge accordingly
- check integrity.

How do user and system share the work?
How does the system cope with not-so-nice situations?

Supporting Conceptualization

The case of the unbound variable:
color(yellow) -> disturbed(liver)   color(x) -> disturbed(z)
relation(yellow, liver) & color(yellow) -> disturbed(liver)

The case of the missing argument: where is the patient?
color(x) & patient(y) -> color2 (x,y)
color2(yellow,tim), color2(pale,john), color2(red,mary)

The case of the interesting constants:
color2(yellow,x) -> yellow(x)
disturbed2(liver,x) -> disturbed-liver(x)
yellow(x) -> liver-disturbed(x)

The case of the missing predicate:
exceptions of a rule used for concept formation (Wrobel 88)

Input Devices

support the user in constructing the domain model

Interface with focussing capabilities, graphics, specialized windows, input scratchpad, ...
Topology of predicates giving an overview of the domain as a task structure
Sort taxonomy showing the objects of the domain grouped together as they play a role in certain relations or share properties

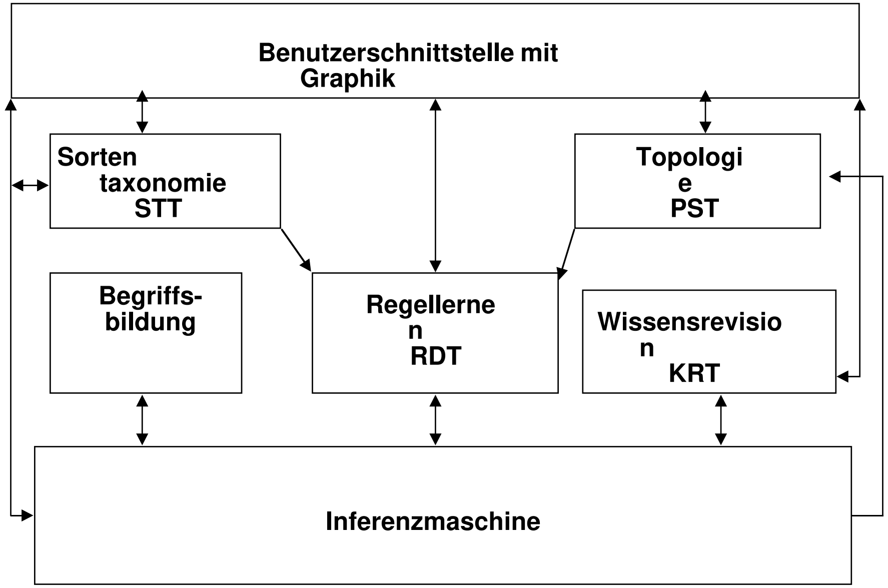
support machine learning in restricting the hypothesis space

machine only learns along the topology structure, and where objects are already grouped together in the sort taxonomy

restrictions are themselves partially learned by the machine

sort taxonomy is learned by the machine from regular user input

MOBAL



Experience

The constructive view of knowledge and knowledge acquisition allows

- to identify machine learning as knowledge acquisition
- to unify the framework for machine learning and manual knowledge acquisition
- to support the user constructing and revising the represented domain model
- to support the user by the machine constructing and revising parts of it

The challenge remains
- to make domain model and represented domain model the same
- to support all aspects and all phases of the modeling cycle