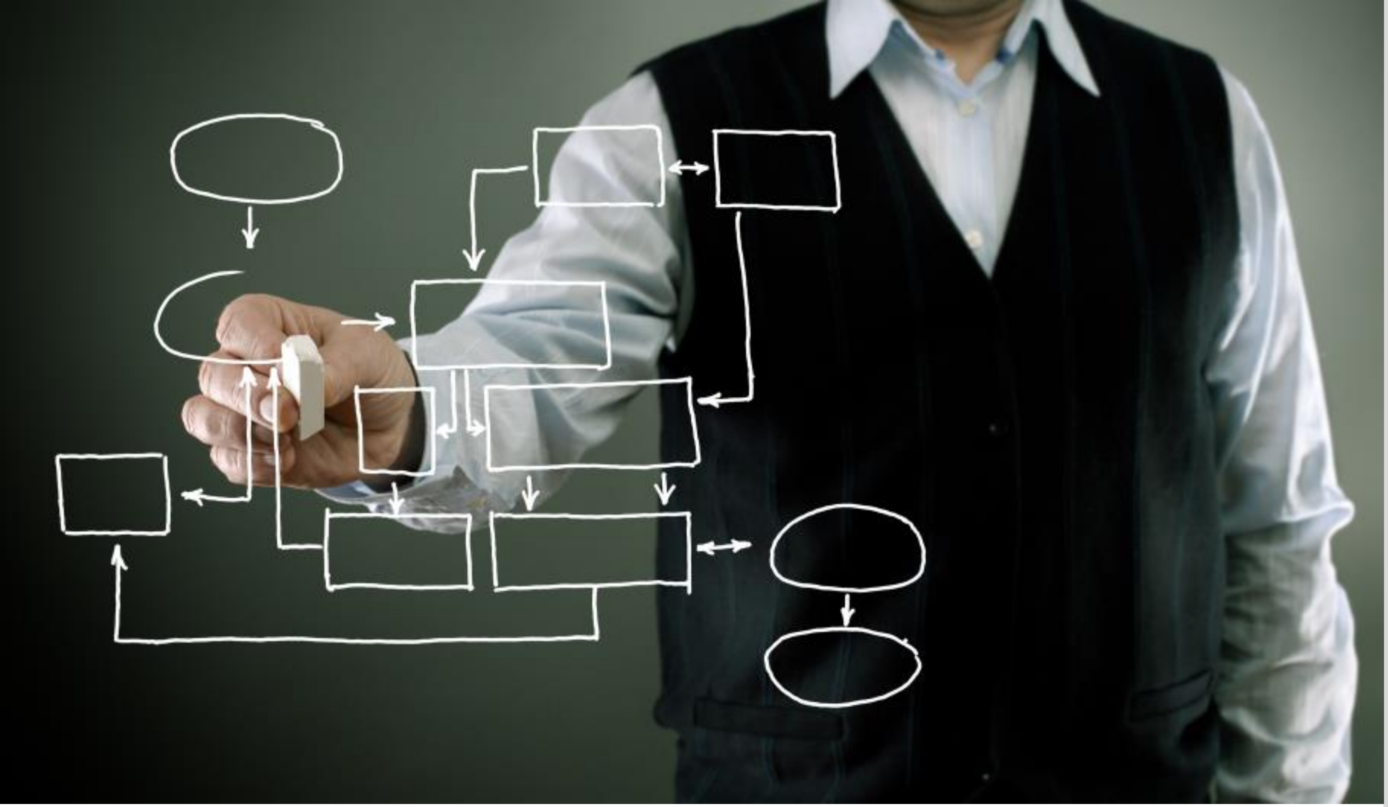


OBERÖSTERREICH

HeuristicLab

Josef Ressel-Zentrum HEUREKA!

Concept Learning and Version Space

HEAL

HEURISTIC AND EVOLUTIONARY ALGORITHMS LABORATORY

Concept Learning and Version Space

Hypothesis space H:

- Determines the success of the learning process
- Background knowledge can be used
- The more complex, the bigger the risk of overfitting
- Probably doesn't contain the best function

- Learning = Search of "good" Hypothesis in H

- Example:

Class of polynomial of degree m:

$$\mathcal{H} = \left\{ x \mapsto \sum_{i=0}^{m} \alpha_i \cdot x^i \mid \alpha_0, \alpha_1, \ldots, \alpha_m \in \Re \right\}$$

Problem of binary Classification:

e.g. $Y=\{0,1\}$ or $Y=\{-1,1\}$

The instances $\{x \in X \mid class(x)=1\}$ can be seen as extensions of concept.

The target is the description of a concept

e.g. dog $\Leftrightarrow$ four legs, fur, tail, can bark

Example:

- Instances are persons, described with attributes
- height ∈ [150, 200] weight ∈ [50,100]
- Target concept is X
- Given positive und negative examples

height	weight	X-Typ
170	77	0
175	72	1
186	66	0
195	90	1
...	...	...

Concept Learning and Version Space

- If H is the Hypothesis space, then every element $h \in H$ is a transformation $h: X \rightarrow \{0,1\}$
- A Hypothesis is **consistent** with given data $S=\{(x_1,y_1)\ldots\ldots(x_n,y_n)\}$ **iff**

$$\forall\, (i \in [1,n])\; h(x_i) = \mathrm{class}(x_i) = y_i.$$

- A version space regarding H and S is defined as set of all consistent Hypothesis:

$$V_{H,S} := \{h \in H \mid h \text{ is consistent with } S\}$$

Concept Learning and Version Space

The version space learning algorithm:

1. Initialize the version space V as a list of all hypothesis $h \in H$.
2. Eliminate for all examples $(x_i, y_i) \in S$ all hypothesis h of V where $h(x_i) \neq y_i$
3. Output V (List of consistent hypothesis)

Question: Representation of V

HeuristicLab OBERÖSTERREICH

A Hypothesis h_2 is more general than h_1 and h_1 is more specific than h_2, $h_1 \lhd h_2$, iff

$$(h_1 \unlhd h_2) \land \neg(h_2 \unlhd h_1)$$

with

$$h_1 \unlhd h_2 \Leftrightarrow_{\text{def}} \{ x \mid h_1(x) = 1 \} \subseteq \{ x \mid h_2(x) = 1 \}$$

$\unlhd$ defines a complete lattice where the biggest element is $h_\top \equiv 1$ and the smallest $h_\perp \equiv 0$

A lattice is an algebraic structure (L, $\lor$, $\land$), consisting of a set L and two binary operations $\lor$, and $\land$, on L is a lattice if the following axiomatic identities hold for all elements a, b, c of L.

- Commutative laws
- Associative laws
- Absorption laws

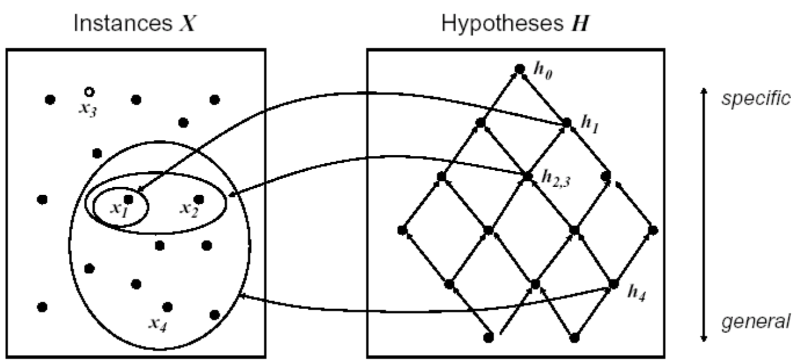
An ordering can be defined as follows:

$v \leq w$, iff $v \cap w = v$.

Concept Learning and Version Space

Find-S Algorithm:

1. **Initialize *h* to the most specific hypothesis in *H***
2. **For each positive training instance *x***
    1. For each attribute constraint a_i in h:
        1. If the constraint a_i in h is satisfied by *x* Then do nothing
        2. Else replace a_i in *h* by the next more general constraint that is satisfied by *x*
3. **Output hypothesis *h***
    1. Guaranteed to output the most specific hypothesis within *H* that is consistent with the positive training examples.
    2. Notice that negative examples are ignored.



	h_0 = < {} {} {} {} {} {} >
x_1 = < sunny warm normal strong warm same >, +	h_1 = < sunny warm normal strong warm same >
x_2 = < sunny warm high strong warm same >, +	h_2 = < sunny warm ? strong warm same >
x_3 = < rainy cold high strong warm change >, –	h_3 = < sunny warm ? strong warm same >
x_4 = < sunny warm high strong cool change >, +	h_4 = < sunny warm ? strong ? ? >

Candidate Elimination Algorithm

$G \leftarrow$ set of maximally general hypotheses in H
$S \leftarrow$ set of maximally specific hypotheses in H

For each training example x do

(a) if x is a **positive** example of concept c:

- Remove from G any hypothesis inconsistent with (i.e., that does not match) x
- For each hypothesis s in S that is not consistent with x:
  - Remove s from S
  - Add to S all *minimal* generalisations h of s such that
    1. h is consistent with x
    2. there is some hypothesis in G that is a generalisation of h
  - Remove from S any hypothesis that is a generalisation of another hypothesis in S

(b) if x is a **negative** example of concept c:

- Remove from S any hypothesis inconsistent with (i.e., that erroneously matches) x
- For each hypothesis g in G that is not consistent with x:
  - Remove g from G
  - Add to G all *minimal* specialisations h of g such that
    1. h is consistent with x
    2. there is some hypothesis in S that is a specialisation of h
  - Remove from G any hypothesis that is a specialisation of another hypothesis in G

Example for Candidate Elimination:

Training examples:

x_1 = <	sunny	warm	normal	strong	warm	same	>, +
x_2 = <	sunny	warm	high	strong	warm	same	>, +
x_3 = <	rainy	cold	high	strong	warm	change	>, –
x_4 = <	sunny	warm	high	strong	cool	change	>, +

Version Space:

S_0: { < {} {} {} {} {} {} > }

S_1: { < sunny warm normal strong warm same > }

S_3 = S_2: { < sunny warm ? strong warm same > }

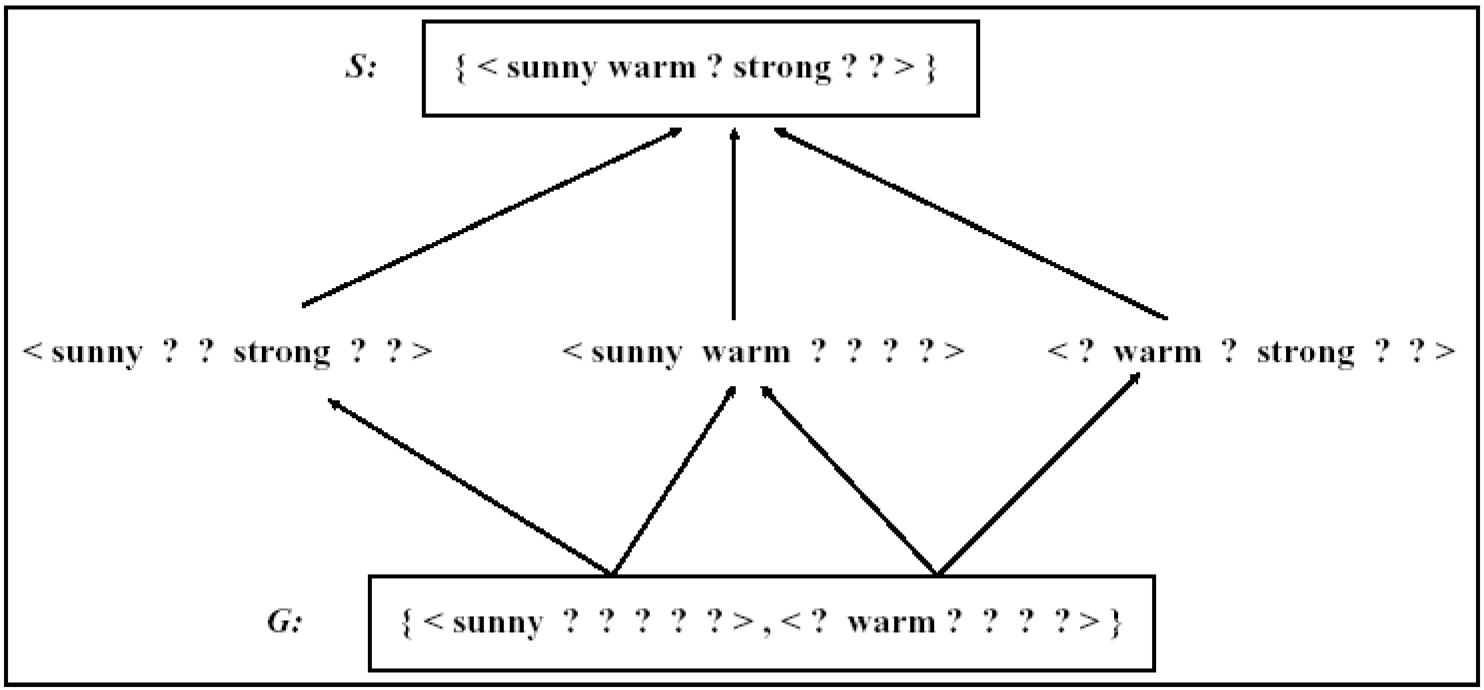
S_4: { < sunny warm ? strong ? ? > }

G_4: { < sunny ? ? ? ? ? >, < ? warm ? ? ? ? > }

G_3: { < sunny ? ? ? ? ? >, < ? warm ? ? ? ? >, < ? ? ? ? ? same > }

G_2 = G_1 = G_0: { < ? ? ? ? ? ? > }

Classification with version space



< sunny warm normal strong cool change >, ? +

< rainy cool normal light warm same >, ? –

< sunny warm normal light warm same >, ? might be + or –

Properties of Candidate Elimination

- Detects if concept has been learn,
  if S = G = a set with only one hypothesis
- Detects, if trainings data is inconsistent:
  if version space is empty (S = G = {})
- Considers all consistent hypothesis equal:
  parallel execution of all possible generalizations and specializations
- Can be used to classify new data:
  - New instances fulfill all hypothesis in S
    - x fulfills all hypothesis in version space
    - x must be classified positive
  - New instance fulfills no hypothesis in G $\Rightarrow$ negative
  - Else: x cannot be classified for sure