Concept Learning

- concept

some subset of objects or events defined over a large set

example.

the subset of animals that constitute birds

representation of concept:

a boolean valued function defined over a large set

example.

a function defined over all animals, whose value is true (1) for birds and false (0) for other animals

- learning

inducing general functions from specific training examples

- concept learning (or category learning)

acquiring the definition of general category given a sample of positive and negative training examples of category, that is, inferring a boolean-valued function from training examples of its input and output

- a concept learning task

. target concept: EnjoySport

(days on which Aldo enjoys water sport)

. hypothesis: a vector of six constraints,

specifying the value of six attributes, they are,

Sky (Sunny/Cloudy/Rainy), AirTemp (Warm/Cold),

Humidity (Normal/High), Wind (Strong/Weak),

Water (Warm/Cool), Forecast (Same/Change)

for each attribute, the hypothesis will either

? (don't care: any value is acceptable),

discrete values, or

∅ (null: no value is acceptable)

example.

<?, Cold, High, ?, ?, ?>

-> "Aldo enjoys sport only on cold days with high humidity."

<?, ?, ?, ?, ?, ?>

-> "Aldo always enjoys sport." (most general hypothesis)

<∅, ∅, ∅, ∅, ∅, ∅>

-> "Aldo does not enjoy sport at all." (most specific case)

. Positive and negative training examples for the target concept EnjoySport

Example	Sky	AirTemp	Humidity	Wind	Water	Forecast	EnjoySport
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

What is *the general concept* for these examples?

Given

instances X: possible days, each described by six attribute,

target function c: EnjoySport $X \rightarrow \{0, 1\}$,

hypothesis H: conjunction of literals such as

<?, Cold, High, ?, ?, ?>, and

training examples D: positive and negative examples of the target function, that is,

$< X_1, c(X_1) >, \cdots, < X_m, c(X_m) >$,

determine

a hypothesis h in H such that

$h(x) = c(x)$ for all x in X.

- inductive learning hypothesis

Any hypothesis found to be approximate the target function well over *a sufficiently large set training examples* will also approximate the target function well over *other unobserved examples*.

- concept learning as search

find a hypothesis that best fits training examples

search space in EnjoySport:

number of instances = $3 \cdot 2^5 = 96$

number of hypotheses = $5 \cdot 4^5 = 5120$

- general-to-specific ordering

. Let $x \in X$ and $h \in H$. Then,

x satisfies h if and only if $h(x) = 1$.

. Let h_j and h_k be boolean-valued functions defined over X. Then,

h_j is *more_general_than_or_equal_to* h_k ($h_j \geqq_g h_k$) if and only if

$(\forall x \in X)[(h_k(x) = 1) \rightarrow (h_j(x) = 1)]$.

h_j is *(strictly) more_general_than* h_k ($h_j >_g h_k$) if and only if

$(h_j \geqq_g h_k) \wedge \neg(h_k \geqq_g h_j)$.

example.

h_j=<Sunny, ?, ?, ?, ?, ?> $>_g$ h_k=<Sunny, ?, ?, Strong, ?, ?>

$\rightarrow$ h_j is more_general_than h_k. or

$\rightarrow$ h_k is more_specific_than h_j.

Here, the problem is how to search the good hypothesis using this hypothesis ordering.

One of such candidates is Find-S algorithm in which
the maximally specific hypothesis is searched.

- Find-S algorithm

Step 1. Initialize h to the most specific hypothesis in H, that is,

h = <∅, ∅, ∅, ∅, ∅, ∅>.

Step 2. For each *positive training instance* x

- for each attribute constraint a_i in h

  if the constraint a_i in h is satisfied by x, do nothing

  else replace a_i in h by the next more general constraint

  that is satisfied by x.

Step 3. Output h.

example.

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>−

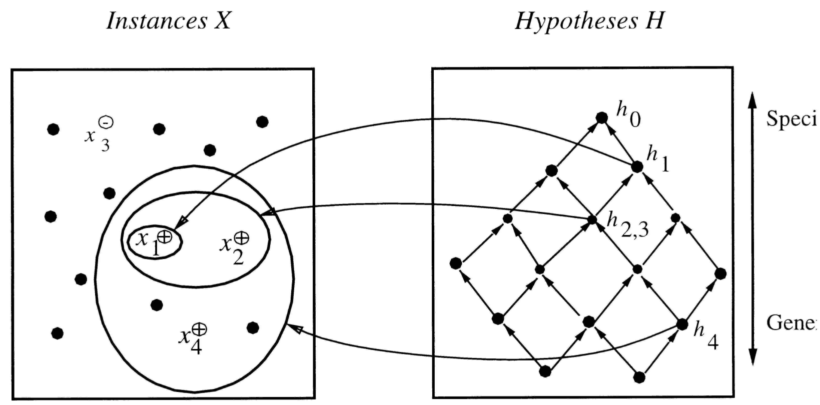
h_3=h_2

x_4=<Sunny, Warm, High, Strong, Cool, Change>+

h_4=<Sunny, Warm, ?, Strong, ?, ?>

… … …

Hypothesis space searched by Find-S algorithm



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*>, +

h_0 = <∅, ∅, ∅, ∅, ∅, ∅>

h_1 = <*Sunny Warm Normal Strong Warm*

h_2 = <*Sunny Warm ? Strong Warm Sar*

h_3 = <*Sunny Warm ? Strong Warm Sam*

h_4 = <*Sunny Warm ? Strong ? ?* >

- problem in Find-S algorithm

. can't tell whether it has the learned concept.
. can't tell when training data are inconsistent.
. picks a maximally specific h.
. depending on H, there might be several.

-> Find-S algorithm only uses the positive examples.
-> We better find *the proper hypothesis space* rather than a specific hypothesis.
-> the concept of version spaces

- version spaces

. A hypothesis h is *consistent* with a set of training examples D of target concept c if and only if $h(x)=c(x)$ for each training example $<x,c(x)>$ in D, that is,

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

. *The version space*, VS_{HD} with respect to hypothesis space H and training examples D, is the subset of hypotheses from H consistent with all training examples in D, that is,

$$VS_{HD} \equiv \{h \in H |\ Consistent(h,D)\}.$$

. representation

The general boundary G of VS_{HD} is the set of its maximally general members, that is,

$$G \equiv \left\{ g \in H \mid Consistent(g, D) \wedge (\neg \exists g' \in H)((g' >_g g) \wedge Consistent(g', D)) \right\}.$$

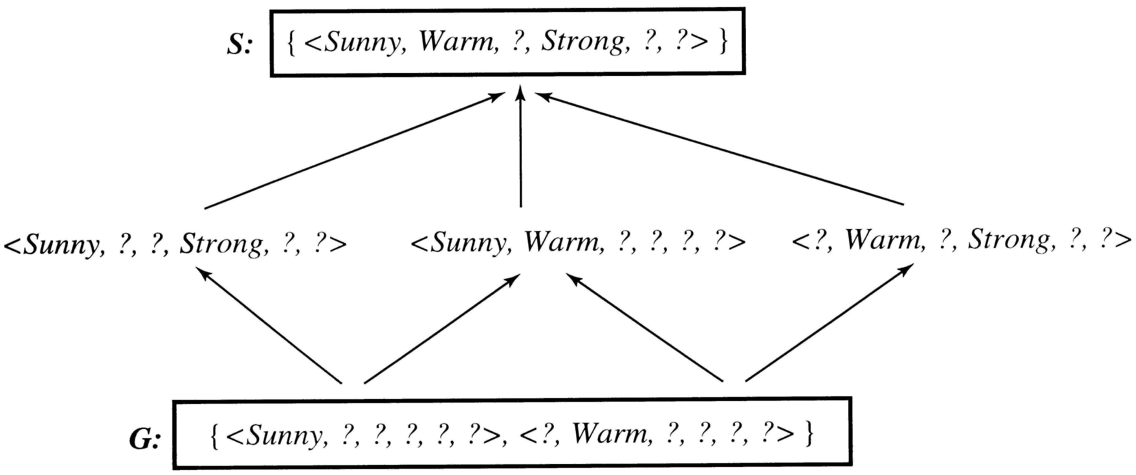
The specific boundary S of VS_{HD} is the set of its maximally specific members, that is,

$$S \equiv \left\{ s \in H \mid Consistent(s, D) \wedge (\neg \exists s' \in H)((s >_g s') \wedge Consistent(s', D)) \right\}.$$

Every member of VS_{HD} lies between these boundaries, that is,

$$VS_{HD} \equiv \left\{ h \in H \mid (\exists s \in S)(\exists g \in G)(g \geqq_g h \geqq_g s) \right\}.$$

Example Version Space



- CE (Candidate Elimination) algorithm

Step 1. Initialize G and S as

$G = \{< ?, ?, ?, ?, ?, ? >\}$ and $S = \{< \varnothing, \varnothing, \varnothing, \varnothing, \varnothing, \varnothing >\}$.

Step 2. For each training sample d, do

- if d is *a positive sample*,
  - (1) remove from G any hypothesis that is inconsistent with d.
  - (2) for each hypothesis s in S that is inconsistent with d,
    - 1) remove s from S.
    - 2) add to S all minimal generalizations h of s such that
      - (i) h is consistent with d, and
      - (ii) some member of G is more general than h.
    - 3) remove from S any hypothesis that is more general than another hypothesis in S.

- if d is *a negative sample*,
  - (1) remove from S any hypothesis that is inconsistent with d.
  - (2) for each hypothesis g in G that is inconsistent with d,
    - 1) remove g from G.
    - 2) add to G all minimal specifications of h of g such that
      - (i) h is inconsistent with d, and
      - (ii) some member of S is more specific than h.
  - (3) remove from G any hypothesis that is less general than another hypothesis in G.

Example Trace (initialize G and S)

S_0: {<Ø, Ø, Ø, Ø, Ø, Ø>}

G_0: {<?, ?, ?, ?, ?, ?>}

Example Trace (Example 1 and 2)

S_0: {<∅, ∅, ∅, ∅, ∅, ∅>}

↓

S_1: {<*Sunny, Warm, Normal, Strong, Warm, Same*>}

↓

S_2: {<*Sunny, Warm, ?, Strong, Warm, Same*>}

G_0, G_1, G_2: {<*?, ?, ?, ?, ?, ?*>}

Training examples:

1. <*Sunny, Warm, Normal, Strong, Warm, Same*>, *Enjoy Sport = Yes*
2. <*Sunny, Warm, High, Strong, Warm, Same*>, *Enjoy Sport = Yes*

Example Trace (Example 3)

S_2, S_3: { <*Sunny, Warm, ?, Strong, Warm, Same*> }

G_3: {<*Sunny, ?, ?, ?, ?, ?*> <*?, Warm, ?, ?, ?, ?*> <*?, ?, ?, ?, ?, Same*>}

G_2: {<*?, ?, ?, ?, ?, ?*> }

Training Example:

3. <*Rainy, Cold, High, Strong, Warm, Change*>, *EnjoySport=No*

Example Trace (Example 4)

S_3: {<*Sunny, Warm, ?, Strong, Warm, Same*>}

S_4: { <*Sunny, Warm, ?, Strong, ?, ?*>}

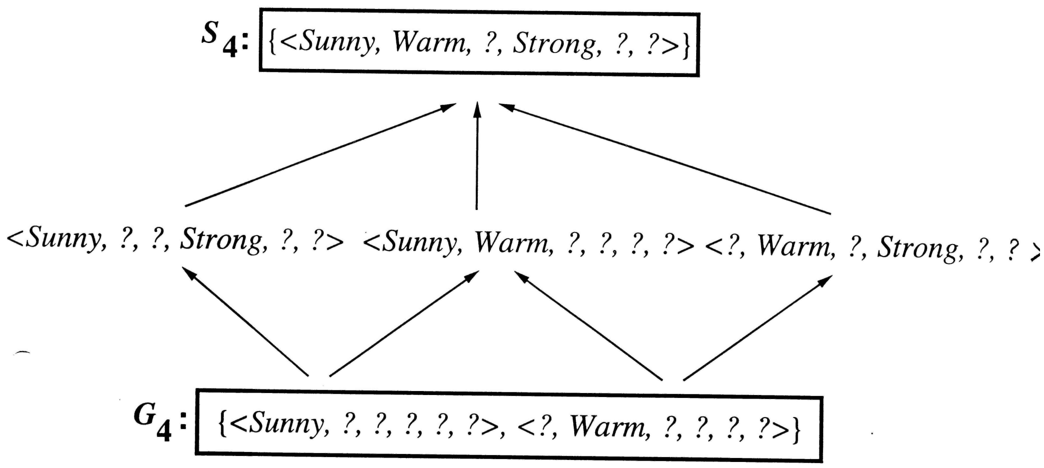
G_4: {<*Sunny, ?, ?, ?, ?, ?*> <*?, Warm, ?, ?, ?, ?*>}

G_3: {<*Sunny, ?, ?, ?, ?, ?*> <*?, Warm, ?, ?, ?, ?*> <*?, ?, ?, ?, ?, Same*>}

Training Example:

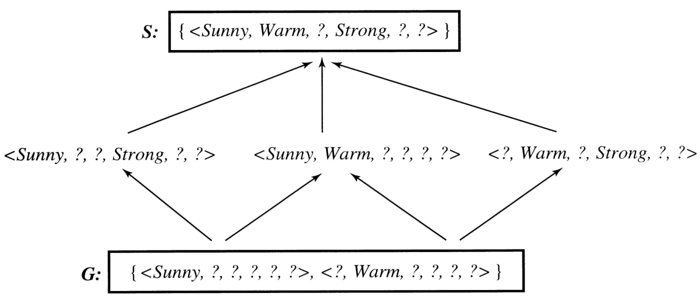
4. <*Sunny, Warm, High, Strong, Cool, Change*>, *EnjoySport = Yes*

Example Trace (The Final Version Space)



The final version space for the *EnjoySport* concept learning problem

How should these be classified?



$\langle Sunny\ Warm\ Normal\ Strong\ Cool\ Change \rangle$ +

$\langle Rainy\ Cool\ Normal\ Light\ Warm\ Same \rangle$ −

$\langle Sunny\ Warm\ Normal\ Light\ Warm\ Same \rangle$?

<Sunny [illegible] Normal Strong Warm Same> −

- CE algorithm will converge toward the hypothesis that correctly describes the target concept, provided

  (1) *no errors in training examples (no noise)*

  (2) *target concept is included in the hypothesis space* H.

- **inductive bias**

  . In EnjoySport, H contains *only conjunction* of attribute values, that is, the disjunctive target concepts such as

  $$< Sunny, ?, ?, ?, ?, ? > \vee < Cloudy, ?, ?, ?, ?, ? >$$

  can not be described.

  . If $H^{'}$ contains conjunction, disjunction, negation over H,

  $|H^{'}| \gg |H|$ $\rightarrow$ large number of samples are required to generalize hypotheses due to large version space.

  example (EnjoySport):

  $|X| = 3 \cdot 2^5 = 96$ distinctive instances

  $|H| = 5 \cdot 4^5 = 5120$ syntactically distinctive hypotheses

  or $1 + 4 \cdot 3^5 = 973$ semantically distinctive hypotheses

  $|H^{'}| = 2^{|X|} = 2^{96} \approx 10^{28}$ distinctive hypotheses

  . A learner that makes no apriori assumptions regarding the identity of the target space has no rational basis for classifying any unseen instances.

  So we need *some assumption on* H. $\rightarrow$ inductive bias

. inductive inference

Let

L : an arbitrary learning algorithm,

C : an arbitrary target concept,

$D_c = < x, c(x) >$: an arbitrary set of training data, and

$L(x_i, D_c)$: classification that L assigns to x_i (new instance) after learning D_c.

Then, inductive inference step performed by L is described by

$(D_c \wedge x_i) \succ L(x_i, D_c)$.

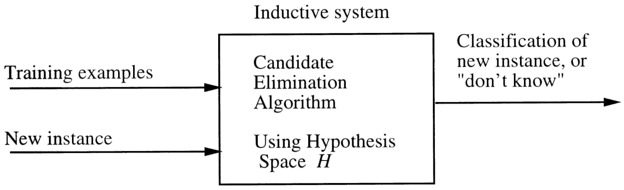
→ $L(x_i, D_c)$ *is inductively inferred from* $(D_c \wedge x_i)$.

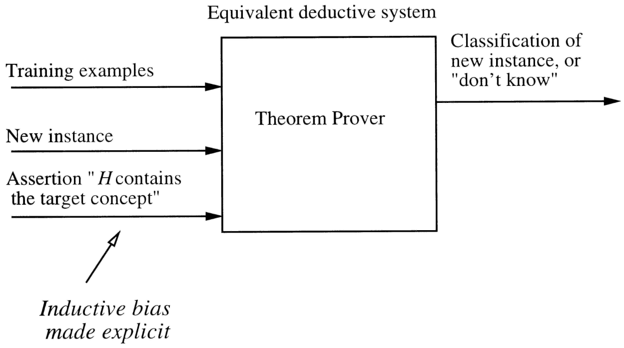
. The inductive bias of L is *any minimal set of assertion* B such that for any target concept c and corresponding training examples D_c

$(\forall x_i \in X)((B \wedge D_c \wedge x_i) \vdash L(x_i, D_c))$

→ for all x_i, $L(x_i, D_c)$ *follows deductively from* $(B \wedge D_c \wedge x_i)$ or we can say that $L(x_i, D_c)$ *is provable from* $(B \wedge D_c \wedge x_i)$.

- inductive bias and equivalent deductive system





- examples of inductive bias

. Rote learner: store examples, classify x if and only of it matches previously observed samples $\rightarrow$ *no inductive bias*.

. CE algorithm: *the target concept* c *is contained in the given hypothesis space* H, that is, $c \in H$. Because, if $c \in H$, the inductive inference performed by CE algorithm can be proved deductively:

(1) $c \in H \vdash c \in VS_{HD_c}$.

(2) $L(x_i, D_c)$ is defined to be the unanimous vote of all hypotheses in VS_{HD_c}.

(3) Therefore, $c(x_i) = L(x_i, D_c)$.

. Find-S algorithm:

(1) $c \in H$

(2) All instances are negative instances unless the opposite is entailed by its other knowledge. This implies that *only the positive instances are meaningful* for the target concept.

Reference: T. Mitchell, “Machine Learning,” Chapter 2.