Lecture 4
Concept Learning

1. **What is a concept learning task**
2. **Find-S algorithm**
3. **Find the version space – List then elimination**

1. Introduction

- Many learning involves acquiring general concepts from specific training examples.
- Each concept can be thought of as a Boolean-valued function defined over a set (e.g. a function defined over all animals, whose value is true for birds and false for other animals).
- **Concept learning** is approximating a Boolean-valued function from examples.

A Concept Learning Task

Sky	Temp	Humid	Wind	Water	Forecst	EnjoySpt
Sunny	Warm	Normal	Strong	Warm	Same	Yes
Sunny	Warm	High	Strong	Warm	Same	Yes
Rainy	Cold	High	Strong	Warm	Change	No
Sunny	Warm	High	Strong	Cool	Change	Yes

- What is the general concept?

A Concept Learning Task - cont

- Many possible representations
- Here, *h* is conjunction of constraints on attributes
- Each constraint can be
  - a specific value (e.g., *Water* = *Warm)*
  - don't care (e.g., *"Water* = ? ")
  - no value allowed (e.g., "Water = ø")

For example,

Sky	AirTemp	Humid	Wind	Water	Forecst
<*Sunny*	?	?	*Strong*	?	*Same*>

A Concept Learning Task - cont

- **Given:**
  - Instances X: Possible days, each described by the attributes *Sky*, *AirTemp*, *Humidity*, *Wind*, *Water*, *Forecast*
  - Target function c: $EnjoySport : X \rightarrow \{0,1\}$
  - Hypotheses H: Conjunctions of literals. E.g.

$$\langle ?, Cold, \mathrm{High}, ?, ?, ? \rangle \;.$$

  - Training examples D: Positive and negative examples of the target function

$$\langle x_1, c(x_1) \rangle \;, \ldots \; \langle x_m, c(x_m) \rangle$$

The inductive learning hypothesis:

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

General-to-specific Ordering of Hypothesis

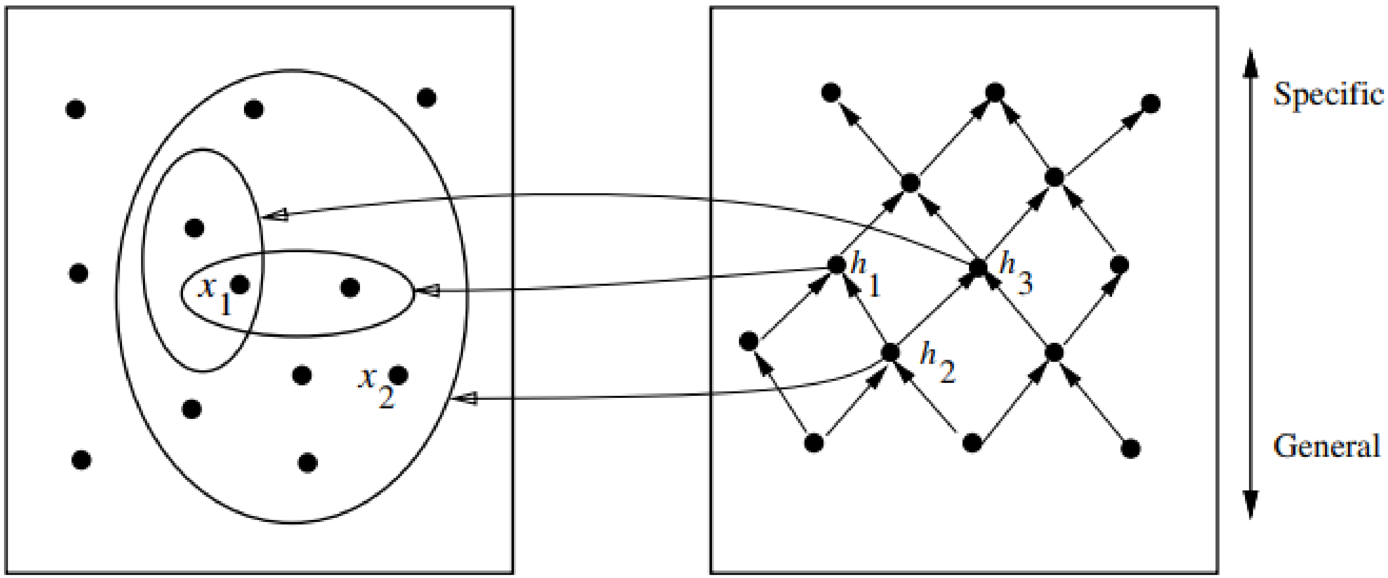
$$h_1 = \langle Sunny, ?, ?, Strong, ?, ? \rangle$$

$$h_2 = \langle Sunny, ?, ?, ?, ?, ? \rangle$$

- **Definition**: 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 (written $h_j \geq_g h_k$ 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 (written $h_j >_g h_k$) if and only if $(h_j \geq_g h_k) \wedge (h_k \not\geq_g h_j)$

Instance, Hypotheses, and More-General-Than

Instances X *Hypotheses H*



x_1 = <*Sunny, Warm, High, Strong, Cool, Same*>
x_2 = <*Sunny, Warm, High, Light, Warm, Same*>

h_1 = <*Sunny, ?, ?, Strong, ?, ?*>
h_2 = <*Sunny, ?, ?, ?, ?, ?*>
h_3 = <*Sunny, ?, ?, ?, Cool, ?*>

2. Find-S Algorithm

- Initialize *h* to the most specific hypothesis in *H*
- 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*

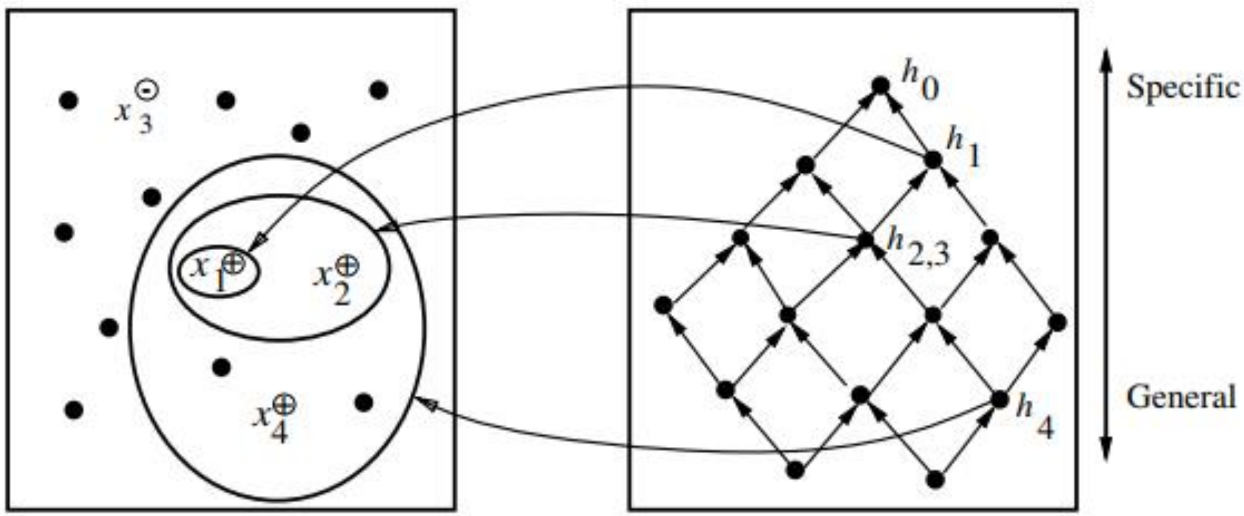
    Then do nothing

    Else replace a_i in *h* by the next more general constraint that is satisfied by *x*

- Output hypothesis *h*

Hypothesis Space Search by Find-S

Instances X *Hypotheses H*



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 ? ?* >

Complaints about Find-S

1. Can't tell whether it has learned concept: Although Find-S will find a hypothesis consistent with the training data, it has no way to determine whether it has found the only hypotheses in H consistent with the data, or whether there are many other consistent hypotheses as well.
2. Picks a maximally specific *h* (why?): In case there are multiple hypotheses consistent with the training examples, Find-S will find the most specific.

Complaints about Find-S - cont

3 Can't tell when training data inconsistent: In some cases, the training examples will contain at least some error or noise. Such inconsistent sets of training examples can severely misled Find-S, given the fact that it ignores negative examples.

4 Depending on *H*, there might be several!: If there are several maximally specific hypotheses spaces, Find-S should be extended to allow it to backtrack on its choices of how to generalize the hypotheses, to accommodate the possibility that the target concept lies along a different branch of the partial ordering than the branch it has selected.

3. Find the Version Space

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 $\langle x, c(x)\rangle$ in *D*.

$$Consistent(h, D) \equiv (\forall \langle x, c(x)\rangle \in D)\ h(x) = c(x)$$

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

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

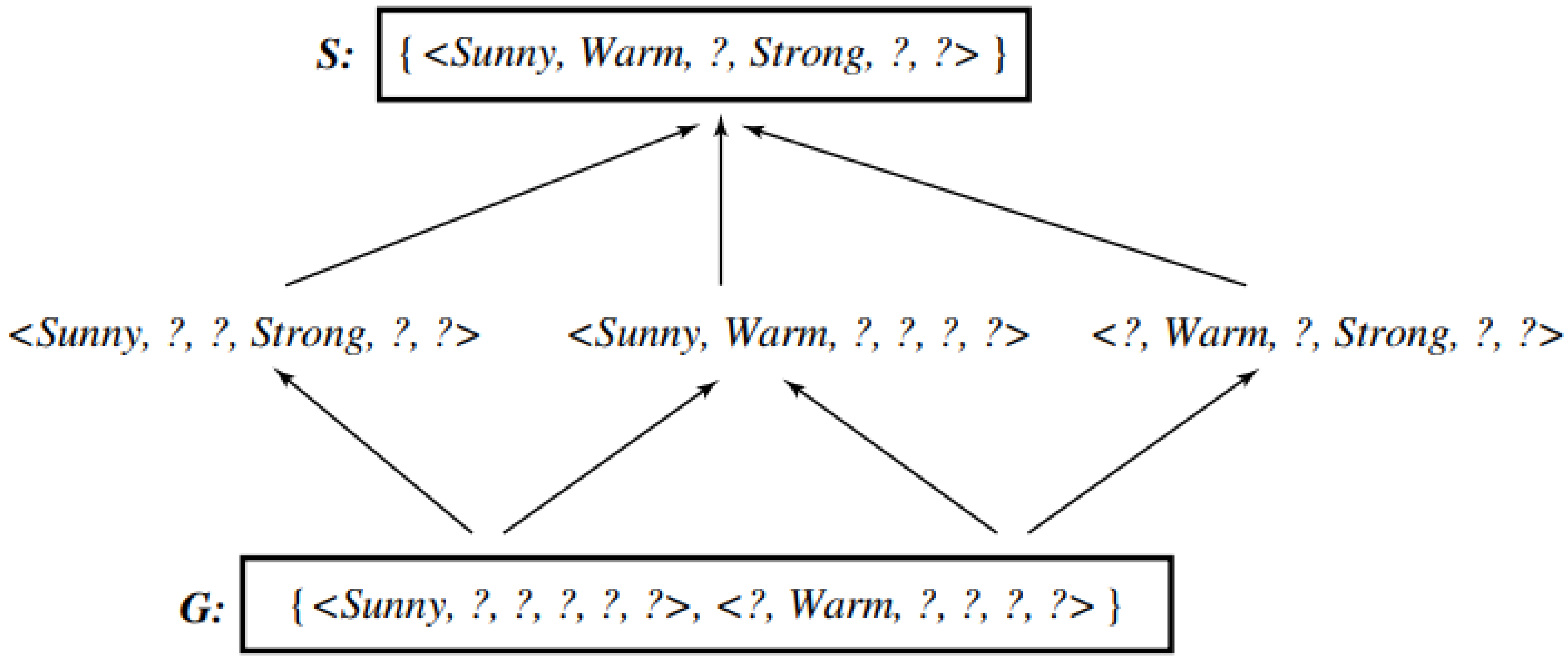
List-Then-Eliminate Algorithm

1. *VersionSpace* — a list containing every hypothesis in H
2. For each training example, $\langle$ x, c(x) $\rangle$

   remove from *VersionSpace* any hypothesis h for which $h(x) \neq c(x)$
3. Output the list of hypotheses in *VersionSpace*

Sky	Temp	Humid	Wind	Water	Forecst	EnjoySpt
Sunny	Warm	Normal	Strong	Warm	Same	Yes
Sunny	Warm	High	Strong	Warm	Same	Yes
Rainy	Cold	High	Strong	Warm	Change	No
Sunny	Warm	High	Strong	Cool	Change	Yes

Example Version Space



List-Then-Eliminate Algorithm - cont

- Recall that Find-S algorithm outputs the hypotheses

  $h = (Sunny, Warm, ?, Strong, ?, ? \rangle$

- In fact, there are six different hypotheses from H that are consistent with these training examples.

- The **Candidate-Elimination algorithm** represents the version space by storing only its most general member (G) and it most specific member (S).

- Given only these two sets S and G, it is possible to enumerate all members of the version space as needed by generating the hypotheses that lie between these two sets in the general-to-specific partial ordering over hypotheses.

Representing Version Spaces

The **General boundary**, G, of version space $VS_{H,D}$ is the set of its maximally general members of H consistent with D.

$$G \equiv \{g \in H | Consistent(g, D) \wedge (\neg \exists g' \in H)[(g' >_g g) \wedge Consistent(g', D)]\}$$

The **Specific boundary**, S, of version space $VS_{H,D}$ is the set of its maximally specific members of H consistent with D.

$$S \equiv \{s \in H | Consistent(s, D) \wedge (\neg \exists s' \in H)[(s >_g s') \wedge Consistent(s', D)]\}$$

Candidate Elimination Algorithm

G maximally general hypotheses in *H*

S maximally specific hypotheses in *H*

For each training example *d*, do

- If *d* is a positive example
  - Remove from *G* any hypothesis inconsistent with *d*
  - For each hypothesis *s* in *S* that is not consistent with *d*
    - Remove s from *S*
    - Add to *S* all minimal generalizations *h* of s such that
      - h is consistent with *d*, and some member of *G* is more general than *h*
    - Remove from *S* any hypothesis that is more general than another hypothesis in *S*

- If *d* is a negative example
  - — Remove from *S* any hypothesis inconsistent with *d*
  - — For each hypothesis *g* in *G* that is not consistent with *d*
    - * Remove *g* from *G*
    - * Add to *G* all minimal specializations *h* of *g* such that

      –h is consistent with d, and some member of *S* is more specific than *h*
    - * Remove from *G* any hypothesis that is less general than another hypothesis in *G*

Example Trace

S_0: $\{<\emptyset, \emptyset, \emptyset, \emptyset, \emptyset, \emptyset>\}$

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

What Next Training Example?

Sky	Temp	Humid	Wind	Water	Forecst	EnjoySpt
Sunny	Warm	Normal	Strong	Warm	Same	Yes
Sunny	Warm	High	Strong	Warm	Same	Yes
Rainy	Cold	High	Strong	Warm	Change	No
Sunny	Warm	High	Strong	Cool	Change	Yes

- S_0: $\{\langle \emptyset, \emptyset, \emptyset, \emptyset, \emptyset, \emptyset \rangle\}$

$\downarrow$

- S_1: $\{\langle Sunny, Warm, Normal, Strong, Warm, Same \rangle\}$

$\downarrow$

- S_2: $\{\langle Sunny, Warm, ?, Strong, Warm, Same \rangle\}$

- G_0, G_1, G_2: $\{\langle ?, ?, ?, ?, ?, ? \rangle\}$

Sky	Temp	Humid	Wind	Water	Forecst	EnjoySpt
Sunny	Warm	Normal	Strong	Warm	Same	Yes
Sunny	Warm	High	Strong	Warm	Same	Yes
Rainy	Cold	High	Strong	Warm	Change	No
Sunny	Warm	High	Strong	Cool	Change	Yes

What Next Training Example?

- S_2, S_3: $\{\langle Sunny, Warm, ?, Strong, Warm, Same\rangle\}$

- G_3:

  $\{\langle Sunny, ?, ?, ?, ?, ?\rangle \langle ?, Warm, ?, ?, ?, ?\rangle \langle ?, ?, ?, ?, ?, Same\rangle\}$

  $\uparrow$

- G_2: $\{\langle ?, ?, ?, ?, ?, ?\rangle\}$

Sky	Temp	Humid	Wind	Water	Forecst	EnjoySpt
Sunny	Warm	Normal	Strong	Warm	Same	Yes
Sunny	Warm	High	Strong	Warm	Same	Yes
Rainy	Cold	High	Strong	Warm	Change	No
Sunny	Warm	High	Strong	Cool	Change	Yes

What Next Training Example?

- S_3: $\{\langle Sunny, Warm, ?, Strong, Warm, Same\rangle\}$

$\downarrow$

- S_4: $\{\langle Sunny, Warm, ?, Strong, ?, ?\rangle\}$

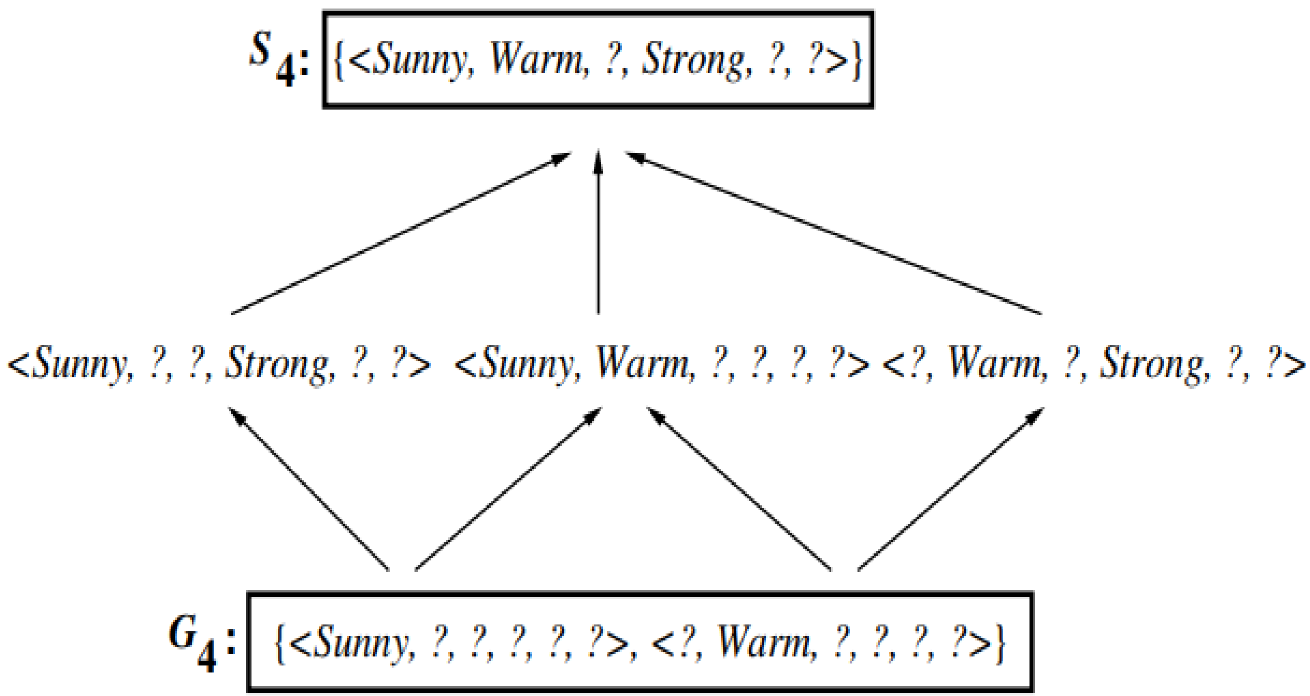
- G_4: $\{\langle Sunny, ?, ?, ?, ?, ?\rangle \langle ?, Warm, ?, ?, ?, ?\rangle\}$

$\uparrow$

- G_3: $\{\langle Sunny, ?, ?, ?, ?, ?\rangle \langle ?, Warm, ?, ?, ?, ?\rangle \langle ?, ?, ?, ?, ?, Same\rangle\}$

Sky	Temp	Humid	Wind	Water	Forecst	EnjoySpt
Sunny	Warm	Normal	Strong	Warm	Same	Yes
Sunny	Warm	High	Strong	Warm	Same	Yes
Rainy	Cold	High	Strong	Warm	Change	No
Sunny	Warm	High	Strong	Cool	Change	Yes

Final Version Space for EnjoySport



Remarks

- The version space learned by the Candidate-elimination algorithm will converge toward the hypotheses that correctly describes the target concept, provided that

1. there are no errors in the training examples, and
2. there is some hypothesis in H that correctly describes the target concept.