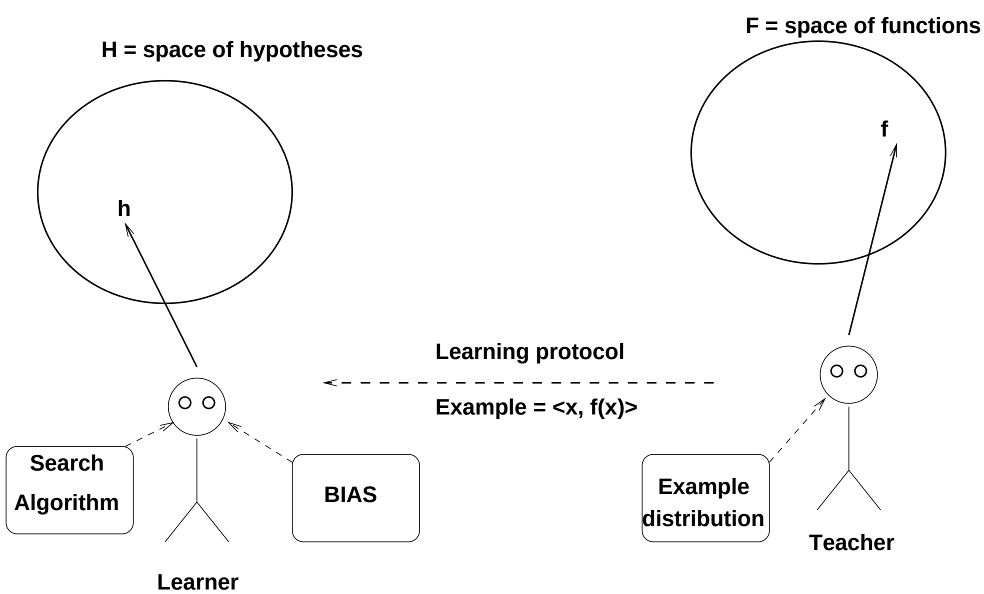
Overview of Week 2

- Concept learning: search in hypotheses space
- Version spaces: candidate elimination algorithm
- Using bias in concept learning

General Model of Concept Learning



Concept Learning

Inferrring a boolean function from labeled training examples.

Example: "user profile" for web browsing:

Dom.	Plat.	Browser	Day	Screen	Cont.	Click?
edu	Mac	Net3	Mon.	XVGA	America	Yes
com	Mac	NetCom	Tue.	XVGA	America	Yes
com	PC	IE	Sat.	VGA	Eur.	No
org	Unix	Net2	Wed.	XVGA	America	Yes

Concept Learning Problem

Given:

- Instances X:
  - *Domain*: edu, com, org
  - *Platform*: Mac, PC, Unix
  - *Browser*: Netscape2, Netscape 3, Netscape Communicator, Microsoft IE.
  - *Day:* Monday - Sunday.
  - *Screen:* VGA or XVGA.
  - *Continent:* America, Europe, Africa, Asia, Australia.

- Hypotheses H: Each $h \in H$ hypotheses is described by a conjunction of constraints on the above attributes (value, ?, ϕ).
- Target concept: Click $c : X \rightarrow 0, 1$
- Training examples D: positive and negative examples of target concept.

Determine: A hypothesis $h \in H$ s.t. $h(x) = c(x) \forall x \in X$.

Hypotheses Space

- Hypotheses language: Every attribute can be a specific value, a wildcard (?), or null (ϕ).
- If an instance i satisfies a hypothesis h, then i is a positive example (else i is a negative example).
- Let X be the set of instances. For the web example, $|X| = 2520$. (why?) How many possible concepts over X?
- Let H denote the set of all hypotheses representable in the hypotheses language.
- For the web example, number of *syntactically* distinct hypotheses is $H = 37800$ (why?)
- For the web example, number of *semantically* distinct hypotheses is $H = 11521$ (why?)

Inductive learning hypotheses

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

Why is this true?

Sampling: Statistical theory for inferring population parameters from samples.

Occam's razor: "Small" hypotheses are likely to be more accurate than larger ones. (e.g. Kepler's law vs. epicycles).

- David Hume: An inquiry concerning human understanding (1748).
- Nelson Goodman: Fact, fiction, and forecast (1979).

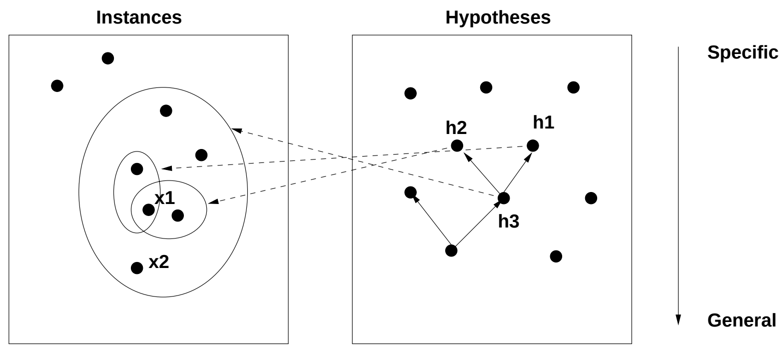
Concept Learning as Search in Hypotheses Space

- The hypotheses can be partially ordered under *more_general_than_or_equal_to* ($\geq_g$).
- $h_1 \geq_g h_2$ iff

$$(\forall x \in X) \quad (h_2(x) = 1) \Rightarrow (h_1(x) = 1)$$

- Example:
  - $h_1 =< edu, Mac, ?, Mon, ?, ? >$
  - $h_2 =< edu, Mac, IE, Mon, ?, Europe >$
- Why is $\geq_h$ a partial ordering?
- Give an example where neither $h_1 \geq_g h_2$ nor $h_2 \geq_g h_1$.

Partial Ordering on Hypotheses Space



x1 = <edu,Mac,IE,Mon,VGA,Eur>

x2 = <edu,PC,IE,Mon,VGA,Eur>

h1 = <edu,Mac,?,?,?,Eur>

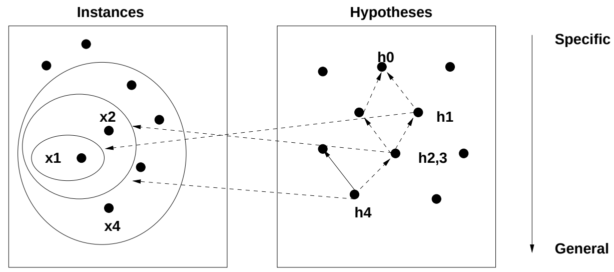
h2 = <edu,?,IE,?,?,Eur>

h3 = <edu,?,?,?,?,Eur>

Find-S: Finding a Maximally Specific Hypothesis

1. Initialize h to the most specific hypothesis in H.
2. For each *positive* instance i, do
   - For each attribute constraint a_i do
     If i is not satisfied by h, then replace a_i by the next more general constraint that is satisfied by i.
3. Output hypothesis h

Example of Find-S



x1 = <edu,Mac,Net3,Mon,XVGA,America>, +

x2 = <com,Mac,Net3,Tue,XVGA,America>, +

x3 = <com,PC,IE,Sat,VGA,Eur>, -

x4 = <org,Unix,Net2,Wed,XVGA,America>, +

h0 = <0,0,0,0,0,0>

h1 = <edu,Mac,Net3,Mon,XVGA,America>

h2 = <?,Mac,Net3,?,XVGA,America>

h3 = <?,Mac,Net3,?,XVGA,America>

h4 = <?,?,?,?,XVGA,America>

Problems with Find-S Algorithm

- Convergence: cannot determine if unique hypothesis
- Singleton hypotheses set: why keep only the most specific h?
- Consistency: what if examples are inconsistent or noisy?
- Multiple specific hypotheses: need not be only one.

Version Space

A hypothesis h is **consistent** with a set of training examples D iff $h(x) = c(x)$ for every $< x, c(x) > \in D$.

The **version space** $VS_{H,D}$ with respect to hypothesis space H and training examples D is the set of all hypotheses $h \in H$ that are consistent with examples in D.

How to compute the version space?

- List-then-eliminate: obvious but impractical idea.
- Candidate elimination (Mitchell, Ph.d. thesis)

Compact Representation of Version Spaces

Key idea: keep only the *boundary* sets, exploiting the partial ordering of the hypotheses space.

General boundary set G: is the set of maximally general members of H consistent with training data D.

$$\{h \in H \mid Consistent(h, D) \wedge (\neg \exists g' \in H) \big((g' >_g h) \wedge Consistent(g', D)\big)\}$$

Specific boundary set S: is the set of maximally specific members of H consistent with training data D.

$$\{h \in H \mid Consistent(h, D) \wedge (\neg \exists g' \in H) \big((h >_g g') \wedge Consistent(g', D)\big)\}$$

Candidate Elimination Algorithm – I

- $G \leftarrow$ the set of maximally general hypotheses in H.
- $S \leftarrow$ the set of 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 s.t. h is consistent with d, and some $g \in G$ is more general than h.
      - Remove from S any hypothesis that is more general than another hypothesis in S.

Candidate Elimination Algorithm – II

- 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 s.t. h is consistent with d, and some $s \in S$ is more specific than h.
    - Remove from G any hypothesis that is less general than another hypothesis in G.

Version Space Example

S0: {<0,0,0,0,0,0>}

G0: {<?,?,?,?,?,?>}

Version Space Example (continued)

S1: {<edu,Mac,Net3,Mon,XVGA,America>}

G1: {<?,?,?,?,?,?>}

<edu,Mac,Net3,Mon,XVGA,America>, +

Version Space Example (continued)

S2: **{<?,Mac,?,?,XVGA,America>}**

G2: **{<?,?,?,?,?,?>}**

<com,Mac,NetCom,Tue,XVGA,America>, +

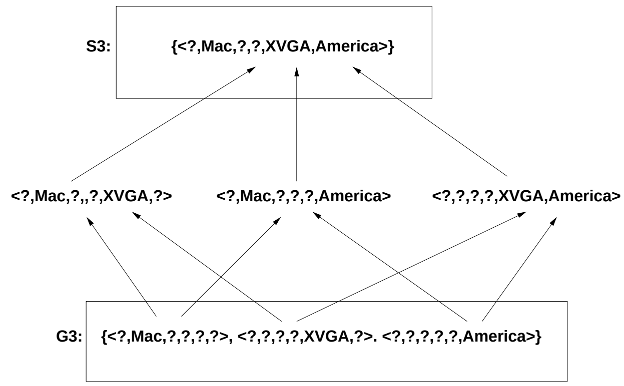
Version Space Example (continued)

S3: **{<?,Mac,?,?,XVGA,America>}**

G3: **{<?,Mac,?,?,?,?>, <?,?,?,?,XVGA,?>. <?,?,?,?,?,America>}**

<com,PC,IE,Sat,VGA,Eur>, -

Active Learning with Version Spaces



What should be the best new example?

Using Partially Learned Concepts

Dom.	**Plat.**	**Browser**	**Day**	**Screen**	**Cont.**	**Click?**
edu	Mac	IE	Fri.	XVGA	America	?
com	PC	NetCom	Wed.	VGA	Europe	?
org	Unix	Net2	Wed.	XVGA	America	?

Version Space Example (continued)

S4: {<?,?,?,?,XVGA,America>}

G4: { <?,?,?,?,XVGA,?>. <?,?,?,?,?,America>}

<org,Unix,Net2,Wed,XVGA,America>, +

Version Space Converged

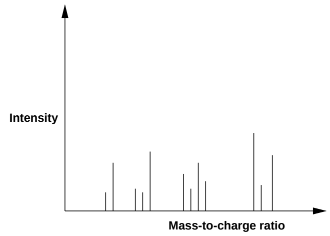
S5: {<?,?,?,?,XVGA,?>}

G5: { <?,?,?,?,XVGA,?>}

<com,Unix,Net2,Wed,XVGA,Europe>, +

Applications of Version Spaces

- META-DENDRAL: Predict molecular structure from mass spectrometer data.



- LEX: Learn heuristics for symbolic integration.

$$\int u dv = uv - \int v du$$

+: $\int 3xcos(x)dx$ with $u = 3x$ and $dv = cos(x)dx$.

-: $\int 5xsin(x)dx$ with $u = sin(x)$ and $dv = 5xdx$.

VS has Exponential Sample Complexity

Let the concept be $A_1 = true$. Let instances be described by n boolean attributes. Consider the sequence of 2^{n-2} examples:

- $A_1 = true \wedge A_2 = true \ldots A_{n-1} = false \wedge A_n = false$
- $A_1 = true \wedge A_2 = true \ldots A_{n-1} = false \wedge A_n = true$
- $A_1 = true \wedge A_2 = true \ldots A_{n-1} = true \wedge A_n = false$
- $A_1 = true \wedge A_2 = true \ldots A_{n-1} = true \wedge A_n = true$

Note that the VS must still contain $A_1 = true$, $A_2 = true$, $A_1 = true \wedge A_2 = true$.

Bias in Concept Learning

- **Bias** is defined as any criteria (other than strict consistency with the training examples) used to select one specific generalization over another.
- *Source of bias:*
  - Hypothesis (generalization) language: (e.g only ? allowed).
  - Generalization algorithm: Find-s.
- What is an unbiased generalization language (algorithm) for the space of instances described by n boolean attributes?

Bias-Free Learning

- Assume H can represent all possible boolean formulae on the attributes (conjunctions, disjunctions, negations).
- Example: (Platform=Macintosh $\vee$ Platform = Unix) $\wedge$ $\neg$ (Platform = PC).
- Given positive examples $x_1, \ldots, x_i$ and negative examples $y_1, \ldots, y_j$, what are the S and G sets?
  - $S = x_1 \vee x_2 \vee \ldots x_i$
  - $G = \neg y_1 \wedge \neg y_2 \ldots \neg y_j$
- Bias-free learning does not allow making inductive leaps beyond the observed training instances!

Bias cannot be eliminated!

- An unbiased generalization algorithm (e.g. version spaces) that uses an unbiased hypothesis space (e.g. all boolean functions) can never go beyond the observed training instances.
- **The power of a learning system follows completely from the appropriateness of its biases**.
- **Machine learning is the study of bias**.
- Useful classes of biases:
  - Factual knowledge of the domain
  - Intended use of the learned generalization
  - Knowledge about source of training data
  - Simplicity and generality
  - Analogy with previously learned generalizations

Probability Distribution on Instances

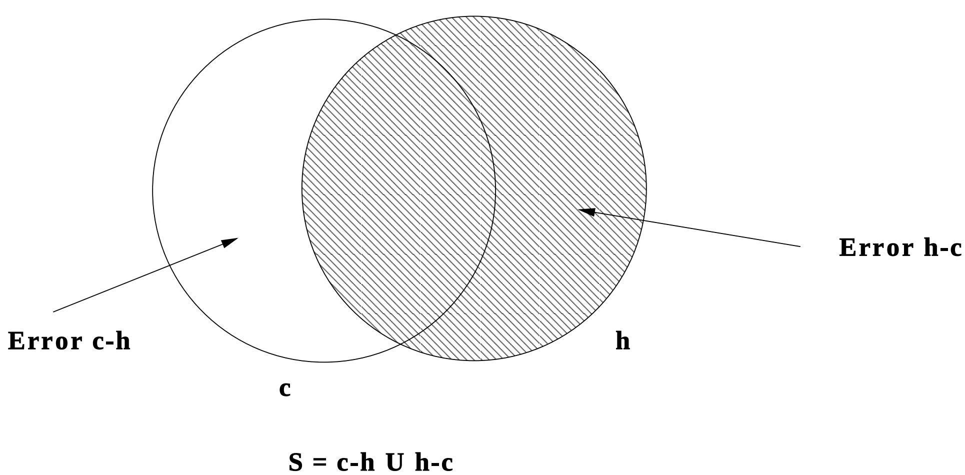
- For any given instance space, there is a **non-uniform** likelihood of seeing different instances. We can represent this situation by imagining that there is a **probability distribution** on the space of instances.
- The learner does not know this distribution ahead of time, but is allowed to assume that it is **fixed**. Thus, a learner trained on one particular distribution should only be tested on that distribution.

Approximate Concept Learning

- Requiring a learner to learn the *right* concept is too strict (e.g. is there a "right" concept of *tree*?).
- Instead, we relax this requirement and allow a learner to produce a **good approximation** to the actual concept.
- Let $P(x)$ be a fixed probability distribution on the instance space. Let c be the target concept, and let h be the concept produced by the learner.
- Let $S = \{x | c(x) \neq h(x)\}$ be the set of instances on which the target concept and the approximation disagree. Let ϵ be an error tolerance parameter where $0 < \epsilon < 1$. Then h is a good approximation (to within ϵ) of c if and only if:

$$\sum_{x \in S} P(x) \leq \epsilon$$

Approximation in Concept Learning



Approximate Learning using Version Spaces

- We say a version space is **exhausted** if the S and G sets are one and the same singleton set. We already know this is too hard.
- Given a hypothesis space H, a target concept c, a sequence of examples Q of c, and an error tolerance ϵ, the version space of Q (w.r.t. H) is ϵ-exhausted if it does not contain any hypothesis that has (**true**) error more than ϵ (w.r.t c).
- We will only require that the learner produce an ϵ-exhausted version space.
- Furthermore, we will solve the problem of exponentially large G sets by simply computing any one hypothesis h that has error $\leq \epsilon$.
- **Question:** How many examples are needed to ϵ-exhaust a version space?

Probabilistic Learning

- Assume training examples are drawn **independently and randomly** from an unknown but fixed distribution P on the instance space.
- We only require that the learner succeed in producing a good approximation to the target concept **with high probability**.
- Specifically, given a confidence parameter δ, we require the learner to be able to ϵ-exhaust a version space with probability at least $1 - \delta$.
- So how many examples are needed for the learner to ϵ-exhaust a version space with probability $\geq 1 - \delta$?

Sample Complexity for Probably Approximate Version Spaces

- **Theorem:** Let H be a finite space of hypotheses, and denote its size by $|H|$. Given m independently drawn random examples (drawn using a fixed distribution P) of some concept c in H, for any $0 < \epsilon < 1$, the probability that the version space consistent with the m examples is not ϵ-exhausted is $\leq |H|e^{-\epsilon m}$.

Proof: Let $h_1, \ldots, h_k$ be hypotheses in H that have error $> \epsilon$.

We will not ϵ-exhaust the version space iff one of these h_i is consistent with all m training examples.

Since each bad hypothesis h_i has error $> \epsilon$, an individual example is consistent with a given bad h_i with probability $\leq 1 - \epsilon$.

The same h_i is consistent with all m examples with probability $\leq (1 - \epsilon)^m$.

Now the probability of any h being consistent with all m examples $\leq k(1 - \epsilon)^m$.

Since $k \leq |H|$, and $(1 - \epsilon)^m < e^{-\epsilon m}$, the result follows.