Knowledge in Learning

Inductive Learning

- Given the set of all hypotheses, eliminate each one that is inconsistent with examples.
  - – The set of all hypotheses is infinite!
  - – Shrink it yet still be realizable

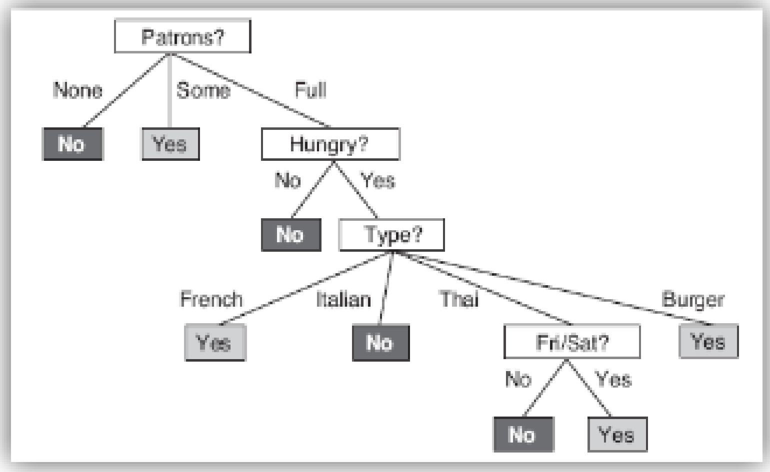
- Current-best-hypothesis search
  - – Maintain a single hypothesis
  - – Generalize it for false negatives
  - – Specialize it for false positives

Hypothesis

Hypotheses, example descriptions, and classification will be represented using **logical sentences**.

Examples

- **attributes** become unary predicates
  $\text{Alternate}(X_1) \land \neg \text{Bar}(X_1) \land \neg\ \text{Fri/Sat}(X_1) \land \text{Hungry}(X_1) \land \ldots$
- **classification** is given by literal using the **goal predicate**
  $\text{WillWait}(X_1)$ or $\neg\ \text{WillWait}(X_1)$



Hypothesis will have the form

$\forall x\ \text{Goal}(x) \Leftrightarrow C_j(x)$

C_j is called the extension of the predicate

$$
\begin{aligned}
\forall r\ \text{WillWait}(r) \Leftrightarrow\ & \text{Patrons}(r,\text{Some}) \\
& \lor (\text{Patrons}(r,\text{Full}) \land \text{Hungry}(r) \land \text{Type}(r,\text{French})) \\
& \lor (\text{Patrons}(r,\text{Full}) \land \text{Hungry}(r) \land \text{Type}(r,\text{Thai}) \land \text{Fri/Sat}(r)) \\
& \lor (\text{Patrons}(r,\text{Full}) \land \text{Hungry}(r) \land \text{Type}(r,\text{Burger}))
\end{aligned}
$$

Hypothesis

Example	Attributes										Goal
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*WillWait*
X_1	*Yes*	*No*	*No*	*Yes*	*Some*	*$$$*	*No*	*Yes*	*French*	*0–10*	*Yes*
X_2	*Yes*	*No*	*No*	*Yes*	*Full*	*$*	*No*	*No*	*Thai*	*3040*	*No*
X_3	*No*	*Yes*	*No*	*No*	*Some*	*$*	*No*	*No*	*Burger*	*0–10*	*Yes*
X_4	*Yes*	*No*	*Yes*	*Yes*	*Full*	*$*	*Yes*	*No*	*Thai*	*10–30*	*Yes*
X_5	*Yes*	*No*	*Yes*	*No*	*Full*	*$$$*	*No*	*Yes*	*French*	*>60*	*No*
X_6	*No*	*Yes*	*No*	*Yes*	*Some*	*$$*	*Yes*	*Yes*	*Italian*	*0–10*	*Yes*
X_7	*No*	*Yes*	*No*	*No*	*None*	*$*	*Yes*	*No*	*Burger*	*0–10*	*No*
X_8	*No*	*No*	*No*	*Yes*	*Some*	*$$*	*Yes*	*Yes*	*Thai*	*0–10*	*Yes*
X_9	*No*	*Yes*	*Yes*	*No*	*Full*	*$*	*Yes*	*No*	*Burger*	*>60*	*No*
X_{10}	*Yes*	*Yes*	*Yes*	*Yes*	*Full*	*$$$*	*No*	*Yes*	*Italian*	*10–30*	*No*
X_{11}	*No*	*No*	*No*	*No*	*None*	*$*	*No*	*No*	*Thai*	*0–10*	*No*
X_{12}	*Yes*	*Yes*	*Yes*	*Yes*	*Full*	*$*	*No*	*No*	*Burger*	*30–60*	*Yes*

Figure 18.3 Examples for the restaurant domain.

Explanation

1. **Alternate:** whether there is a suitable alternative restaurant nearby.
2. **Bar**: whether the restaurant has a comfortable bar area to wait in.
3. **Fri/Sat**: true on Fridays and Saturdays.
4. ***Hungry:*** whether we are hungry.
5. **Patrons**: how many people are in the restaurant (values are None, Some, and Full).
6. **Price:** the restaurant's price range ($, $$, $$$).
7. **Raining:** whether it is raining outside.
8. **Reservation**: whether we made a reservation.
9. **Type**: the kind of restaurant (French, Italian, Thai, or burger).
10. **WaitEstimate**: the wait estimated by the host (0-10 minutes, 10-30, 30-60, >60).

Attribute of Dataset

Learning a rule for deciding whether
To wait for a table.

Examples were described by **attributes** such as *alternate, bar, fri / sat,*

In a logical setting, an example is an object that is described by a logical sentence;
The attributes become unary predicates.
Let us generically call the ith example X_i.
the first example from Figure is described by the sentences

$$Alternate(X_1) \wedge \neg Bar(X_1) \wedge \neg Fri/Sat(X_1) \wedge Hungry(X_1) \wedge \ldots$$

We will use the notation ***Di****(X i)*t o refer to the description of ***Xi,*** wh ere ***Di***c an be any logical expression taking a single argument. The classification of the object is given by the sentence

***Will Wait* (XI)** .

Hypothesis

Generic notation $Q(X_i)$ if the example is positive, and $\neg Q(X_i)$ if the example is negative.

The aim of inductive learning in the logical setting is to find an equivalent logical expression for the goal predicate Q that we can use to classify examples correctly. Each hypothesis proposes such an expression, which we call a **candidate definition** of the goal predicate.

Using C_i to denote the candidate definition, each hypothesis H_i is a sentence of the form

$$\forall x Q(x) \Leftrightarrow C_i(x)$$

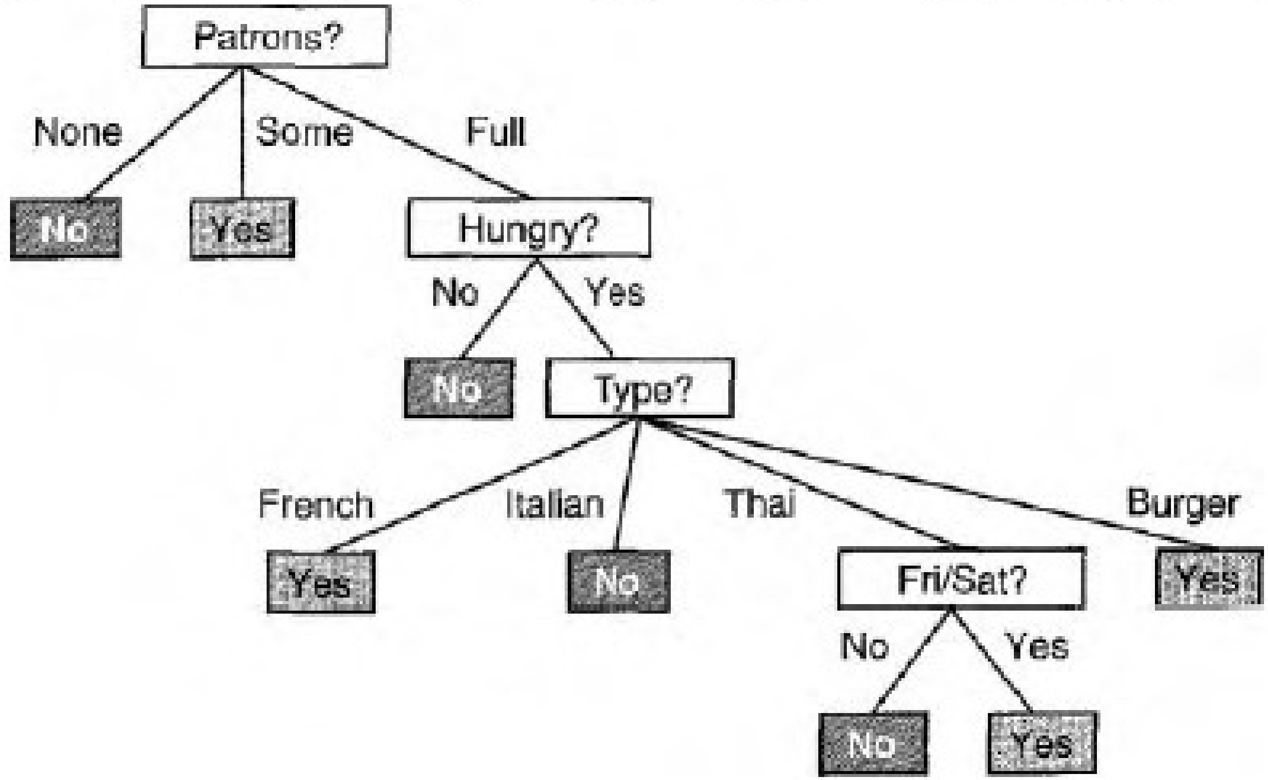


Figure 18.6 The decision tree induced from the 12-example training set.

$$
\begin{aligned}
\forall r \quad WillWazt(r) \Leftrightarrow\ & Patrons(r, Some) \\
\lor\ & Patrons(r, Full) \land Hungry(r) \land Type(r, French) \\
\lor\ & Patrons(r, Full) \land Hungry(r) \land Type(r, Thai) \\
& \quad \land Fri/Sat(r) \\
\lor\ & Patrons(r, Full) \land Hungry(r) \land Type(r, Burger)\,.
\end{aligned}
$$

Hypothesis

Each hypothesis predicts that a certain set of examples.

Set of examples that satisfy a candidate definition = **extension** of the respective hypothesis

Two hypotheses with different extensions are therefore logically inconsistent with each other, because they disagree on their predictions for at least one example.

If they have the same extension, they are logically equivalent.

Hypothesis space

The **hypothesis space** used by a machine learning system is the set of all hypotheses that might possibly be returned by it.

The hypothesis space H is the set of all hypothesis

The **hypothesis space** used by a machine learning system is the set of all hypotheses that might possibly be returned by it.

The hypothesis space **H** is the set of all hypothesis

$\{H_1, H_2, H_3, H_4, \ldots\ldots, H_n,\}$ that the learning algorithm is designed to entertain.

DECISION-TREE-LEARNING algorithm can entertain any decision tree hypothesis defined in terms of the attributes provided; it!;

Hypothesis space therefore consists of all these decision trees. Presumably, the learning algorithm believes that one of the hypotheses is correct; that is, it believes the sentence

DECISION-TREE-LEARNING algorithm can entertain any decision tree hypothesis defined in terms of the attributes provided; it!;

Hypothesis space therefore consists of all these decision trees. Presumably, the learning algorithm believes that one of the hypotheses is correct; that is, it believes the sentence

$$H_1 \vee H_2 \vee H_3, \vee, \ldots\ldots, \vee H_n$$

As the examples arrive, hypotheses that are not **consistent** with the examples can be ruled out.

As the examples arrive, hypotheses that are not **consistent** with the examples can be ruled out.

Learning logicial descripciones

- The process of constructing a decision tree can be seen as searching the **hypothesis space H.** The goal is to construct an hypothesis ***H*** that explains the data in the training set.
- The hypothesis *H* is a logical description of the form:

$$H: \quad \boxed{\begin{array}{l} H_1 \lor H_2 \lor \ldots \lor H_n \\ \forall x\, Q(x) <=> C_i(x) \end{array}}$$

$$H_i:$$

- where ***Q(x)*** is the goal predicate and $\boldsymbol{C_i(x)}$ are candidate definitions.

Learning logicial descripciones

- if hypothesis H_i is consistent with the entire training set, it has to be consistent with each example.
- What would it mean for it to be inconsistent with an example? This can happen in one of two ways:

Two Case:

- **False negative example:** the hypothesis predicts it should be a negative example but it is in fact positive
- **False positive example:** should be positive but it is actually negative

- if hypothesis H_i is consistent with the entire training set, it has to be consistent with each example.
- What would it mean for it to be inconsistent with an example? This can happen in one of two ways:

Two Case:

- **False negative example:** the hypothesis predicts it should be a negative example but it is in fact positive
- **False positive example:** should be positive but it is actually negative

Learning logicial descripciones

An example can be a **false negative** for the hypothiesis, if the hypothesis says it should be negative but in fact it is positive. For instance, the new example *XI3* described by

$$Patrons(X_{13}, Full) \wedge Wait(X_{13}, 0\text{-}10) \wedge \neg Hungry(X_{13}) \wedge \ldots \wedge WillWait(X_{13})$$

would be a false negative for the hypothesis H_r, given earlier. From H_r, and the example description,

- we can deduce both *WillWait(X13)*, which is what the example says,
- $\neg$*WillWait(X13)*, which is what the hypothesis predicts.

The hypothesis and the example are therefore logically inconsistent.

Hypothesis and Hypothesis Space

If an example is a false positive or false negative for a hypothesis, then the example and the hypothesis are logically inconsistent with each other. Assuming that the example is a correct observation of fact, then the hypothesis can be ruled out.

Suppose, for example, that the example is denoted by the sentence I_1, and the hypothesis space is $H_1 \vee H_2 \vee H_3, \vee, \ldots\ldots, \vee H_n$

Then if I_1 is inconsistent with *H2* and *H3*, the logical inference system can deduce the new hypothesis space $H_1 \vee H_4$

Definitions

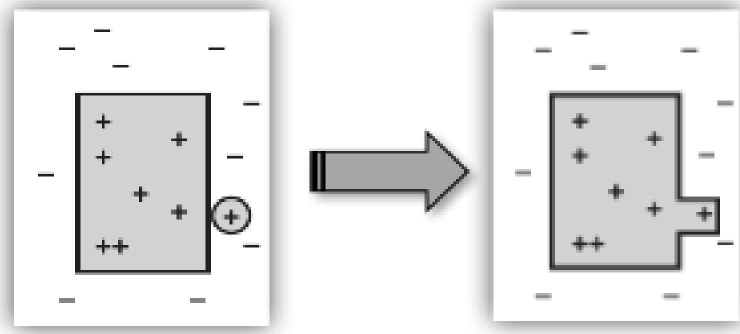
The idea is to maintain a single hypothesis, and to adjust it as new examples arrive in order to maintain consistency

if the example is **consistent** with the hypothesis

then do **not change** it

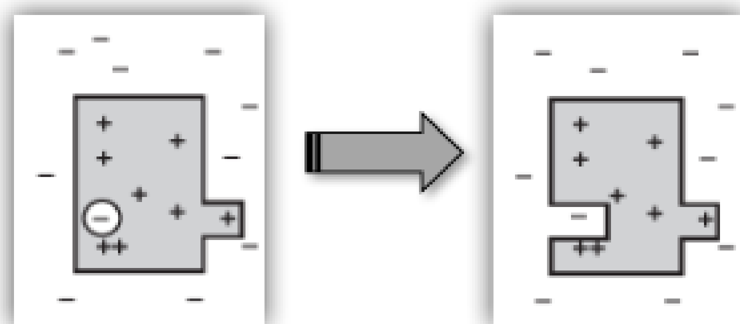
if **false negative**

then **generalize** the hypothesis



if **false positive**

then **specialize** the hypothesis



- Learning algorithm believes that one of its hypotheses is true, i.e. $H_1 \vee H_2 \vee H_3 \vee \ldots$
- Each false positive/false negative could be used to rule out inconsistent hypotheses from the hyp. space
  ➡ general model of inductive learning
- But not practicable if hyp. space is vast, e.g. all formulae of first-order logic
- Have to look for simpler methods:
  - Current-best hypothesis search
  - Version space learning

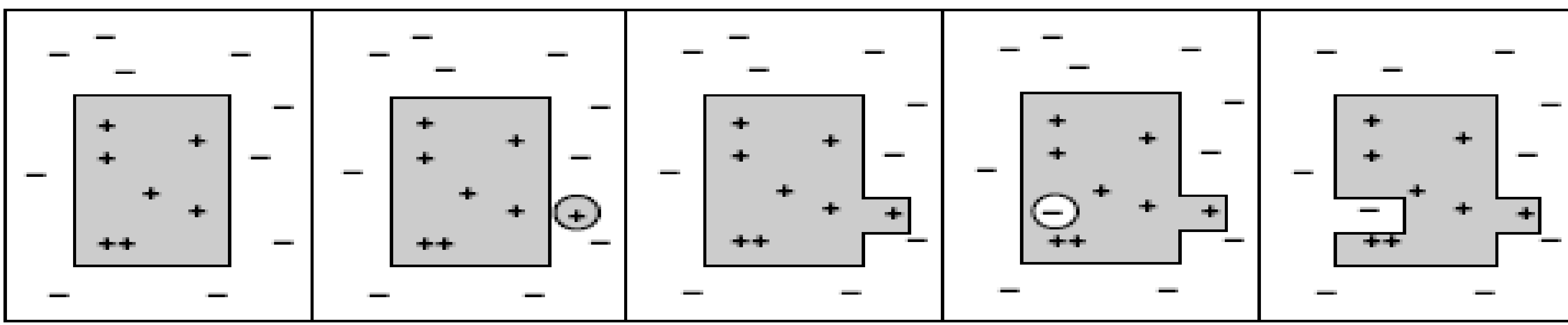
Current-Best Hypothesis Search

- **Idea very simple:** adjust hypothesis to maintain consistency with examples and maintain a single hypothesis
- Uses specialization/generalisation of current hypothesis to exclude false positives/include false negatives.
- Assumes “more general than” and “more specific than” relations to search hypothesis space efficiently

Current-Best Hypothesis Search

- Suppose we have some hypothesis such as H_r,
- Which we have grown quite fond.
- As long as each new example is consistent, we need do nothing.

- Then along comes a false negative example X_{13}.
- What do we do? Figure shows H_r schematically
- As a region: everything inside the rectangle is part of the extension of H_r,
- Examples that have Actually been seen so far are shown as "+" or "-",
- and we see that H_r correctly categorizes all the examples
- as positive or negative examples of *Will Wait*.

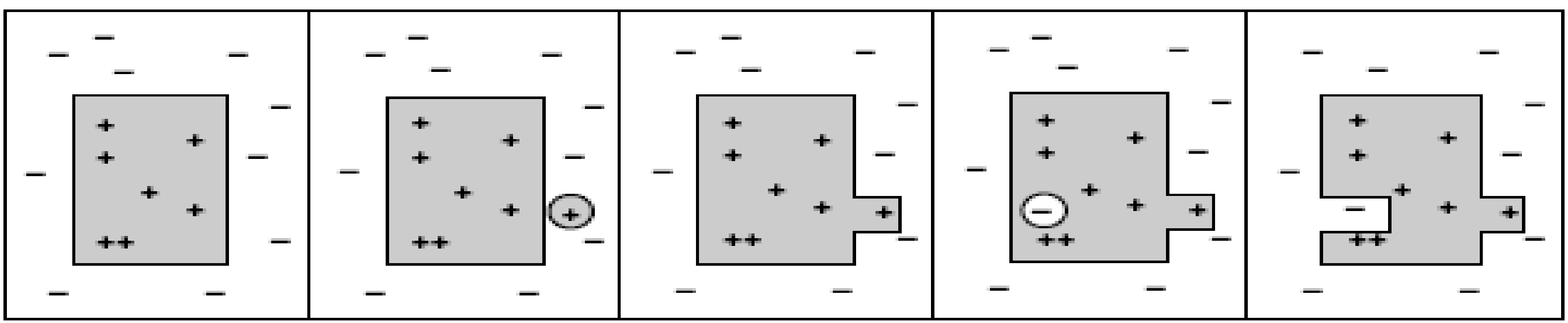


(a) (b) (c) (d) (e)

Current-Best Hypothesis Search

- In Figure 19.l(b), (circled) is a false negative: the hypothesis says
- it should be negative but it is actually positive.
- The extension of the hypothesis must be increased to include it.
- This is called **generalization;** one possible generalization is
- shown in Figure 19.l(c).

- Then in Figure 19.l(d), we see **a false positive**: the hypothesis says the
- **new example (circled) should be positive**, but it **actually is negative**
- The **extension of the hypothesis** must be decreased to exclude the
- example. This is called **specialization;**
- in Figure 19.l(e) we see one possible specialization of the hypothesis.



(a) (b) (c) (d) (e)

Current-Best Hypothesis Search

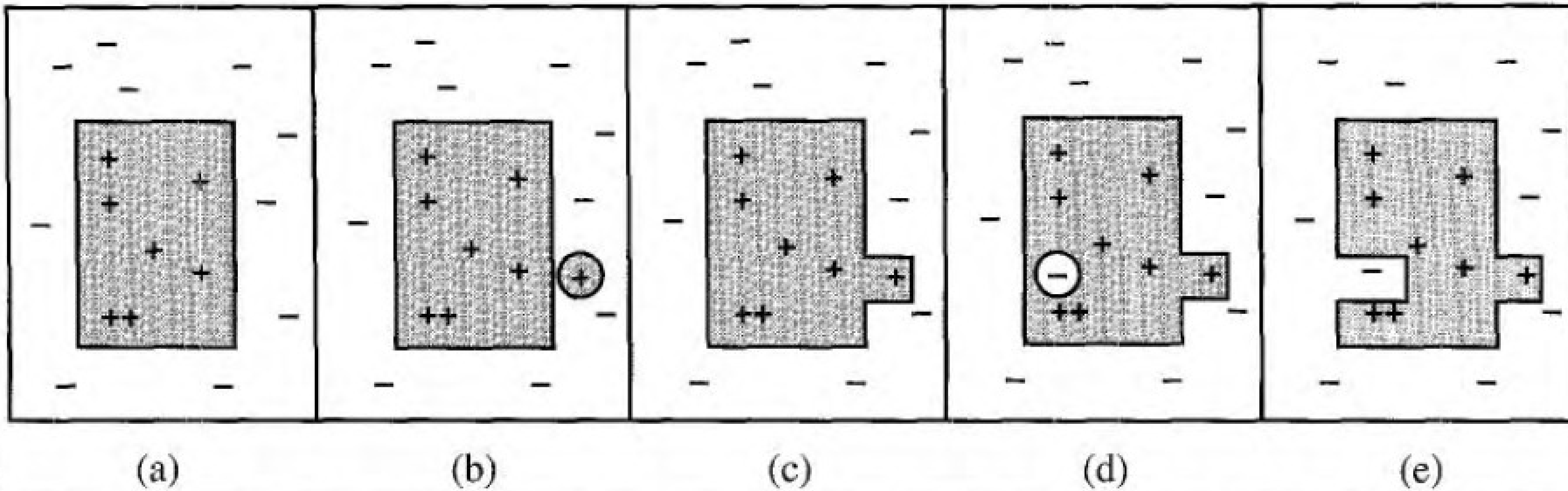


Figure 19.1 (a) A consistent hypothesis. (b) A false negative. (c) The hypothesis is generalized. (d) A false positive. (e) The hypothesis is specialized.

The "more general than" and "more specific than" relations between hypotheses provide the logical structure on the hypothesis space that makes efficient search possible.

We can now specify the CURRENT-B EST-LEARNING algorithm, shown in Figure 19.2. Notice that each time we consider generalizing or specializing the hypothesis, we must check for consistency with the other examples, because an arbitrary increase/decrease in the extension might include/exclude previously seen negative/positive examples.

Current Best Hypothesis

```
function CURRENT-BEST-LEARNING(examples) returns a hypothesis

  H ← any hypothesis consistent with the first example in examples
  for each remaining example in examples do
     if e is false positive for H then
        H ← choose a specialization of H consistent with examples
     else if e is false negative for H then
        H ← choose a generalization of H consistent with examples
     if no consistent specialization/generalization can be found then fail
  return H
```

Figure 19.2 The current-best-hypothesis learning algorithm. It searches for a consistent hypothesis and backtracks when no consistent specialization/generalization can be found.

Things to note:

- Non-deterministic choice of specialization/generalisation
- Does not provide rules for spec./gen.
- **One possibility:** add/drop conditions

Current-best-hypothesis Search

1. Pick a random example to define the initial hypothesis
2. For each example,
   - In case of a false negative:
     - Generalize the hypothesis to include it
   - In case of a false positive:
     - Specialize the hypothesis to exclude it
3. Return the hypothesis

Current-best-hypothesis search

- We have defined generalization and specialization as operations that change the ***extension*** of a hypothesis.
- Now we need to determine exactly how they can be implemented as syntactic operations that change the candidate definition associated with the hypothesis,
- So that a program can carry them out.
- This is done by first noting that generalization and Specialization are also logical relationships between hypotheses

Current-best-hypothesis search

If hypothesis H_1, with definition C_1, is a generalization of hypothesis H_2 with definition C_2, then we must have

$$\forall x\, C_2(x) \Rightarrow C_1(x)$$

Therefore in order to construct a generalization of H_2, we simply need to find a definition C_1 that is logically implied by C_2. This is easily done. For example, if $C_2(x)$ is $Alternate(x) \wedge Patrons(x, some)$ then one possible generalization is given by $C_1(x) = Patrons(x, some)$.

This is called **dropping condition.**

- Intuitively, it generates a weaker definition and therefore allows a larger set of positive examples.
- There are a number of other generalization operations, depending on the language being operated on.
- Similarly, we can specialize a hypothesis by adding extra conditions to its candidate definition or by removing disjuncts from a disjunctive definition.
- Let us see how this works on the restaurant example, using the data in Figure 18.3.

Hypothesis

Example	Attributes										Goal
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*WillWait*
X_1	*Yes*	*No*	*No*	*Yes*	*Some*	*$$$*	*No*	*Yes*	*French*	*0–10*	*Yes*
X_2	*Yes*	*No*	*No*	*Yes*	*Full*	*$*	*No*	*No*	*Thai*	*3040*	*No*
X_3	*No*	*Yes*	*No*	*No*	*Some*	*$*	*No*	*No*	*Burger*	*0–10*	*Yes*
X_4	*Yes*	*No*	*Yes*	*Yes*	*Full*	*$*	*Yes*	*No*	*Thai*	*10–30*	*Yes*
X_5	*Yes*	*No*	*Yes*	*No*	*Full*	*$$$*	*No*	*Yes*	*French*	*>60*	*No*
X_6	*No*	*Yes*	*No*	*Yes*	*Some*	*$$*	*Yes*	*Yes*	*Italian*	*0–10*	*Yes*
X_7	*No*	*Yes*	*No*	*No*	*None*	*$*	*Yes*	*No*	*Burger*	*0–10*	*No*
X_8	*No*	*No*	*No*	*Yes*	*Some*	*$$*	*Yes*	*Yes*	*Thai*	*0–10*	*Yes*
X_9	*No*	*Yes*	*Yes*	*No*	*Full*	*$*	*Yes*	*No*	*Burger*	*>60*	*No*
X_{10}	*Yes*	*Yes*	*Yes*	*Yes*	*Full*	*$$$*	*No*	*Yes*	*Italian*	*10–30*	*No*
X_{11}	*No*	*No*	*No*	*No*	*None*	*$*	*No*	*No*	*Thai*	*0–10*	*No*
X_{12}	*Yes*	*Yes*	*Yes*	*Yes*	*Full*	*$*	*No*	*No*	*Burger*	*30–60*	*Yes*

Figure 18.3 Examples for the restaurant domain.

Current-best-hypothesis search

- The first example X_1 is positive. *Alternate()* (X_1) is true, so let the initial hypothesis be
  $H_1 : \forall x\ WillWait(x) \Leftrightarrow Alternet(x)$.

- The Second example X_2 is negative. H_1 predicts it to be positive, so it is a false positive. Therefore, we need to specialize H_1. This can be done by adding an extra condition that will rule out X_2. One possibility is
  $$H_2 : \forall x\ WillWait(x) \Leftrightarrow Alternet(x) \wedge Patrons(x, some).$$

- The Third example X_3 is positive. H_2 predicts it to be negative, so it is a false negative. Therefore, we need to generalize H_2. *We dropd Alternet* condition.
  $$H_3 : \forall x\ WillWait(x) \Leftrightarrow Patrons(x, some).$$

The fourth example X_4 is positive. H_3 predicts it to be negative, so it is a false negative. Therefore, we need to generalize H_3. We cannot drop the *Patrons* condition, because that would yield an all-inclusive hypothesis that would be inconsistent with X_2. One possibility is to add a disjinct:

that would yield an all-inclusive hypothesis that would be inconsistent with . One possibility is to add a disjinct:

$$H_4 : \forall x\ WillWait(x) \Leftrightarrow Patrons(x, some) \vee \left(Patrons(x, Full) \wedge Fri/Sat(x)\right)$$

Already, the hypothesis is starting to look reasonable. Obviously, there are other possibilities consistent with the first four examples.

Already, the hypothesis is starting to look reasonable. Obviously,

Current-best-hypothesis search

- Here are two of them:

$$H_4' : \ \forall x \ \ WillWait(x) \Leftrightarrow \neg WaitEstimate(x, 30\text{-}60) \, .$$

$$\begin{aligned} H_4'' : \ \forall x \ \ WillWait(x) \Leftrightarrow \ & Patrons(x, Some) \\ & \lor (Patrons(x, Full) \land WaitEstimate(x, 10\text{-}30)) \, . \end{aligned}$$

The CURRENT-BEST-LEARNING algorithm is described non-deterministically, because at any point, there may be several possible specializations or generalizations that can be applied. The choices that are made will not necessarily lead to the simplest hypothesis, and may lead to an unrecoverable situation where no simple modification of the hypothesis is consistent with all of the data. In such cases, the program must backtrack to a previous choice point

Current-best-hypothesis search

- Some difficulties arise:

1. Checking all the previous instances over again for each modification is very expensive.

2. The search process may involve a great deal of backtracking. Hypothesis space can be a doubly exponentially large place.

How to Generalize

- <u>Replacing Constants with Variables:</u>
  Object(Animal,Bird) ← *Object (X,Bird)*
- <u>Dropping Conjuncts:</u>
  Object(Animal,Bird) & Feature(Animal,Wings)
  ← *Object(Animal,Bird)*
- <u>Adding Disjuncts:</u>
  Feature(Animal,Feathers) ←
  Feature(Animal,Feathers) v
  Feature(Animal,Fly)
- <u>Generalizing Terms:</u>
  Feature(Bird,Wings) ← *Feature(Bird,Primary-*
  Feature)

How to Specialize

- <u>Replacing Variables with Constants:</u>
  Object (X, Bird) ← Object(Animal, Bird)
- <u>Adding Conjuncts</u>:
  Object(Animal,Bird) ← Object(Animal,Bird) & Feature(Animal,Wings)
- <u>Dropping Disjuncts</u>:
  Feature(Animal,Feathers) v Feature(Animal,Fly) ← Feature(Animal,Fly)
- <u>Specializing Terms</u>:
  Feature(Bird,Primary-Feature) ← Feature(Bird,Wings)

Generalize and Specialize

- **Must be consistent** with all other examples

- **Non-deterministic**

  –At any point there may be several possible specializations or generalizations that can be applied

How to implement specialization and generalization of the hypothesis?

How to implement specialization and generalization of the hypothesis?

- If hypothesis h_1 is a **generalization** of hypothesis h_2, then we must have
  $\forall x\, C_2(x) \Rightarrow C_1(x)$
- C_i is typically a conjunction of predicates
  - generalization can be realized by **dropping conditions** or by **adding disjuncts**
  - specialization can be realized by **adding extra conditions** or by **removing disjuncts**

Example	Attributes										Target
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*Wait*
X_1	T	F	F	T	Some	$$$	F	T	French	0–10	T
X_2	T	F	F	T	Full	$	F	F	Thai	30–60	F
X_3	F	T	F	F	Some	$	F	F	Burger	0–10	T
X_4	T	F	T	T	Full	$	F	F	Thai	10–30	T
X_5	T	F	T	F	Full	$$$	F	T	French	>60	F
X_6	F	T	F	T	Some	$$	T	T	Italian	0–10	T
X_7	F	T	F	F	None	$	T	F	Burger	0–10	F
X_8	F	F	F	T	Some	$$	T	T	Thai	0–10	T
X_9	F	T	T	F	Full	$	T	F	Burger	>60	F
X_{10}	T	T	T	T	Full	$$$	F	T	Italian	10–30	F
X_{11}	F	F	F	F	None	$	F	F	Thai	0–10	F
X_{12}	T	T	T	T	Full	$	F	F	Burger	30–60	T

A restaurant example:

- the first example is positive, attribute Alternate(X_1) is true, so let the initial hypothesis be
  h_1: $\forall x$ WillWait(x) $\Leftrightarrow$ Alternate(x)
- the second example is negative, hypothesis predicts it to be positive, so it is a false positive; we need to specialize by adding extra condition
  h_2: $\forall x$ WillWait(x) $\Leftrightarrow$ Alternate(x) $\wedge$ Patrons(x,Some)
- the thirst example is positive, the hypothesis predicts it to be negative, so it is a false negative; we need to generalize by dropping the condition Alternate
  h_3: $\forall x$ WillWait(x) $\Leftrightarrow$ Patrons(x,Some)
- The fourth example is positive, the hypothesis predicts it to be negative, so it is a false positive; we need to generalize by adding a disjunct (we cannot drop the Patrons condition)
  h_3: $\forall x$ WillWait(x) $\Leftrightarrow$ Patrons(x,Some) $\vee$ (Patrons(x,Full) $\wedge$ Fri/Sat(x))

Potential Problem of Current-best-hypothesis Search

- Have to check all examples again after each modification
- Involves great deal of backtracking
- Extension made not necessarily lead to the simplest hypothesis.
- May lead to an unrecoverable situation where no simple modification of the hypothesis is consistent with all of the examples.
- The program must backtrack to a previous choice point

Alternative: maintain set of all hypotheses consistent with examples

Potential Problem of Current-best-hypothesis Search

After each modification of the hypothesis we need to **check all the previous examples**.

There are several possible generalizations and specializations and we may need to **backtrack** where no simple modification of the hypothesis is consistent with all the data.

The source of problems – **strong commitment**

- The algorithm has to choose a particular hypothesis as its best guess even though it does not have enough data yet to be sure of the choice.

A solution could be **least-commitment search**.

Least-commitment search

Backtracking arises because the current-best-hypothesis approach has to **choose** a particular hypothesis as its best guess even though it does not have enough data yet to be were of the choice.

Incremental approach such that consistency

is guaranteed without backtracking

• Partial ordering on the hypothesis space

– Generalization/specialization

– G-set, most general boundary

– S-set, most specific boundary

Least-commitment search

The hypothesis space can be viewed as a disjunctive sentence

$$h_1 \vee h_2 \vee h_3 \vee \ldots \vee h_n$$

Hypothesis inconsistent with a new example is removed from the disjunction.

Assuming the original hypothesis space does in fact contain the right answer, the reduced disjunction must still contain the right answer.

The set of hypothesis remaining is called the **version space**.

The version space learning algorithm (also the **candidate elimination** algorithm).

Least-commitment search

Version Space=set of remaining hypothesis

Algorithm:

VERSION-SPACE-LEARNING(example)

1. $V \leftarrow set\ of\ all\ hypothesis$
2. **for each** example e in examples **do**
3. **if V is not empty**
4. **then** $\boldsymbol{V \leftarrow \{h \in V : h\ is\ consistent\ with\ e\}}$
5. **return V**

function VERSION-SPACE-UPDATE (V, *e*) **returns** an updated version space

$V \leftarrow \{h \in V :\ h \text{ is consistent with } e)$

Figure 19.3 The version space learning algorithm. It finds a subset of V that is consistent with the `examples`.

Least-commitment search

- One important property of this approach is that it is ***incremental:*** one never has to go back and reexamine the old examples.

- All remaining hypotheses are guaranteed to be consistent with them anyway.

- It is also a **least-commitment** algorithm because it makes no arbitrary choices

Least-commitment search

Version Space Learning

Advantage:

- Incremental approach (don’t have to consider old examples again)
- Least-commitment algorithm

Problem: How to write down disjunction of all hypotheses?

think of interval notation [1, 2]

Exploit ordering on hypotheses and boundary sets

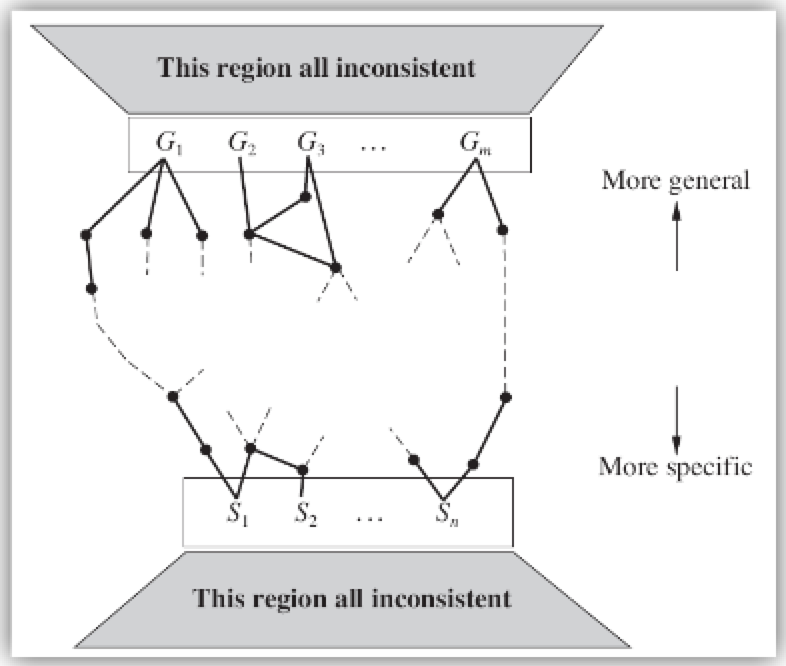
- **G-set** most general boundary (no more general hypotheses are consistent with all examples)

- **S-set** most specific boundary (no more specific hypotheses are consistent with all examples)

Least-commitment search

Hypothesis space is enormous, so how can we possibly write down this enormous disjunction?

We have an ordering of hypothesis space (generalization/specialization) so we can specify boundaries, where each boundary will be a set of hypothesis (a boundary set).



G = a most general boundary

- consistent with all observations so far
- there are no consistent hypotheses that are more general
- initially True

S = a most specific boundary

- consistent with all observations so far
- there are no consistent hypotheses that are more specific
- initially False

Everything in between G-set and S-set is guaranteed to be consistent with the examples and nothing else is consistent.

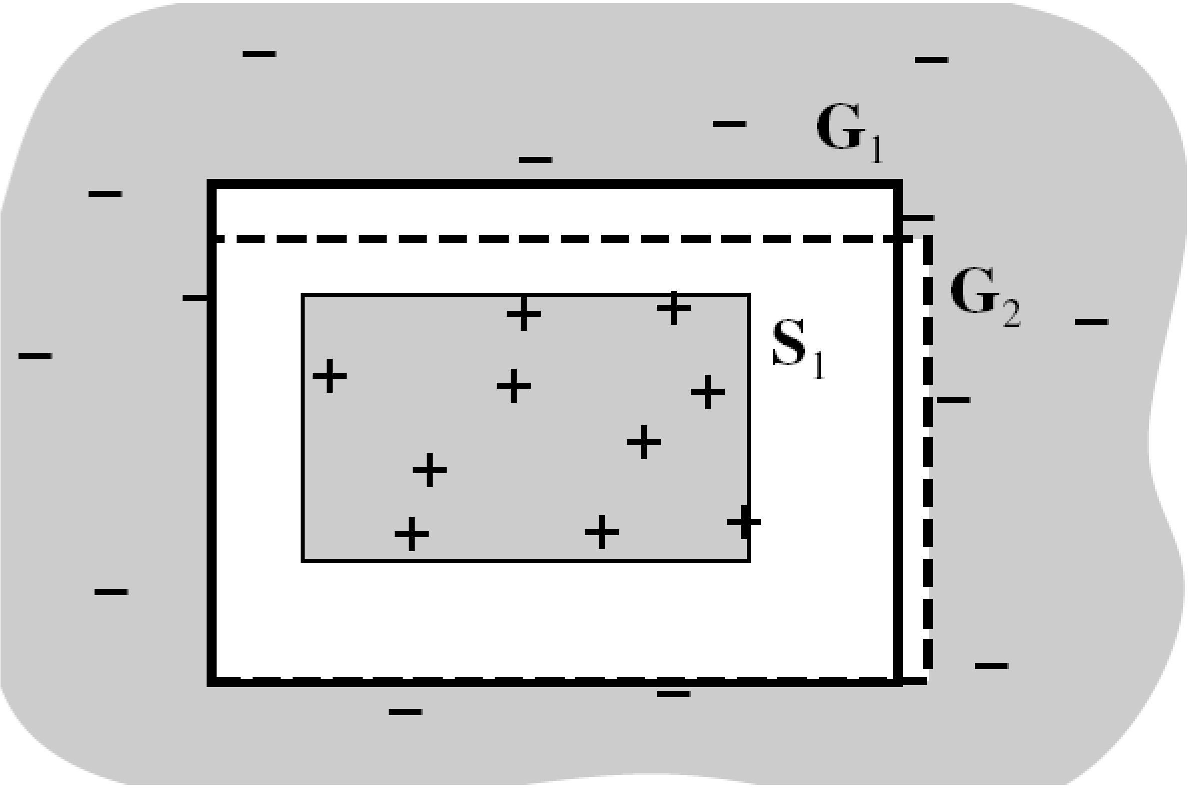
Version Space Learning

- Everything between **G** and **S** (version space) is consistent with examples and represented by boundary sets
- Initially: **G = {True}, S = {False}**
- **How to prove that this is a reasonable representation?**
- Need to show two properties
  - Every consistent **H** not in the boundary sets is more specific than some G_i and more general than some S_i (follows from definition)
  - Every **H** more specific than some G_j and more general than some S_j is consistent.

Any such **H** rejects all negative examples rejected by each member of **G** and accepts all positive examples accepted by any member of $S \rightarrow H$ consistent:

Version Space Learning

There are no known examples "between" S and G, i.e. outside S but inside G:



Summary

- How to deal with knowledge-based representations of inductive learning?
- Described DTL in terms of logic
- Introduced current-best learning (problems: backtracking, non-incremental)
- Version spaces as an incremental method of inductive learning
- Next time: Knowledge Representation & Reasoning