Concept Learning

- Learning from examples
- General-to-specific ordering over hypotheses
- Version Spaces and candidate elimination algorithm
- Picking new examples
- The need for inductive bias
Some Examples for SmileyFaces

<table>
<thead>
<tr>
<th>Eyes</th>
<th>Nose</th>
<th>Head</th>
<th>Fcolor</th>
<th>Hair?</th>
<th>Smile?</th>
</tr>
</thead>
<tbody>
<tr>
<td>● ●</td>
<td>△</td>
<td>O</td>
<td>P</td>
<td></td>
<td>)</td>
</tr>
<tr>
<td>□ □</td>
<td>□</td>
<td>O</td>
<td>G</td>
<td></td>
<td>)</td>
</tr>
<tr>
<td>□ □</td>
<td>△</td>
<td>O</td>
<td>Y</td>
<td></td>
<td>)</td>
</tr>
<tr>
<td>● ●</td>
<td>△</td>
<td>O</td>
<td>G</td>
<td></td>
<td>)</td>
</tr>
<tr>
<td>□ □</td>
<td>□</td>
<td>O</td>
<td>Y</td>
<td></td>
<td>)</td>
</tr>
</tbody>
</table>
# Features from Computer View

<table>
<thead>
<tr>
<th>Eyes</th>
<th>Nose</th>
<th>Head</th>
<th>Fcolor</th>
<th>Hair?</th>
<th>Smile?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Round</td>
<td>Triangle</td>
<td>Round</td>
<td>Purple</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Square</td>
<td>Square</td>
<td>Square</td>
<td>Green</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Square</td>
<td>Triangle</td>
<td>Round</td>
<td>Yellow</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Round</td>
<td>Triangle</td>
<td>Round</td>
<td>Green</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Square</td>
<td>Square</td>
<td>Round</td>
<td>Yellow</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Representing Hypotheses

Many possible representations for hypotheses \( h \)

Idea: \( h \) as conjunctions of constraints on features

Each constraint can be:

- a specific value (e.g., \( Nose = \text{Square} \))
- don’t care (e.g., \( Eyes = ? \))
- no value allowed (e.g., \( Water=\emptyset \))

For example,

\[
\begin{align*}
\text{Eyes} & \quad \text{Nose} & \quad \text{Head} & \quad \text{Fcolor} & \quad \text{Hair} \equiv \\
\text{Round,} & \quad ? & \quad \text{Round,} & \quad ? & \quad \text{No}
\end{align*}
\]
Prototypical Concept Learning Task

Given:

- Instances $X$: Faces, each described by the attributes $Eyes$, $Nose$, $Head$, $Fcolor$, and $Hair$?
- Target function $c$: $Smile? : X \rightarrow \{ \text{no, yes} \}$
- Hypotheses $H$: Conjunctions of literals such as $< ?, Square, Square, Yellow, ? >$
- Training examples $D$: Positive and negative examples of the target function $< x_1, c(x_1) >, < x_2, c(x_2) >, ..., < x_m, c(x_m) >$

Determine: a hypothesis $h$ in $H$ such that $h(x) = c(x)$ for all $x$ in $D$. 
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.

- What are the implications?
- Is this reasonable?
- What (if any) are our alternatives?
- What about concept drift (what if our views/tastes change over time)?
Instances, Hypotheses, and More-General-Than

Instances $X$

$x_1 = \langle \text{Round}, \text{Square}, \text{Square}, \text{Purple}, \text{Yes} \rangle$

$x_2 = \langle \text{Round}, \text{Square}, \text{Round}, \text{Green}, \text{Yes} \rangle$

Hypotheses $H$

$h_1 = \langle \text{Round}, ?, \text{Square}, ?, ?, \rangle$

$h_2 = \langle \text{Round}, ?, ?, ?, \text{Yes} \rangle$

$h_3 = \langle \text{Round}, ?, ?, ?, \rangle$

Hypotheses ordered from specific to general.
**Find-S Algorithm**

1. Initialize $h$ to the most specific hypothesis in $H$

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 $x$ THEN
   
   do nothing

   ELSE

   replace $a_i$ in $h$ by next more general constraint satisfied by $x$

3. Output hypothesis $h$
Hypothesis Space Search by \textit{Find-S}

\[ x_1 = \langle \text{Round, Triangle, Round, Purple, Yes} \rangle + \]
\[ x_2 = \langle \text{Square, Square, Square, Green, Yes} \rangle - \]
\[ x_3 = \langle \text{Square, Triangle, Round, Yellow, Yes} \rangle + \]
\[ x_4 = \langle \text{Round, Triangle, Round, Green, No} \rangle - \]
\[ x_5 = \langle \text{Square, Square, Round, Yellow, Yes} \rangle + \]
\[ h_0 = \langle \phi, \phi, \phi, \phi, \phi \rangle \]
\[ h_1 = \langle \text{Round, Triangle, Round, Purple, Yes} \rangle \]
\[ h_2 = \langle \text{Round, Triangle, Round, Purple, Yes} \rangle \]
\[ h_3 = \langle ?, \text{Triangle, Round, ?, Yes} \rangle \]
\[ h_4 = \langle ?, \text{Triangle, Round, ?, Yes} \rangle \]
\[ h_5 = \langle ?, ?, \text{Round, ?, Yes} \rangle \]
Complaints about \textit{Find-S}

- Cannot tell whether it has learned concept
- Cannot tell when training data inconsistent
- Picks a maximally specific $h$ (why?)
- Depending on $H$, there might be several!

- How do we fix this?
The List-Then-Eliminate Algorithm

1. Set $VersionSpace$ equal to a list containing every hypothesis in $H$

2. For each training example, $<x,c(x)>$
   remove from $VersionSpace$ any hypothesis $h$ for which $h(x) \neq c(x)$

3. Output the list of hypotheses in $VersionSpace$

- But is listing all hypotheses reasonable?
- How many different hypotheses in our simple problem?
  - How many not involving “?” terms?
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 in $D$.

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

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 $D$.

$$VS_{H,D} \equiv \{h \in H \mid \text{Consistent}(h, D)\}$$
Example Version Space

G: \{ <?,?,Round,?,?>, <?,Triangle,?,?,?> \}

S: \{ <?,Triangle,Round,?,Yes> \}

<?,?,Round,?,Yes> \quad <?,Triangle,Round,?,?> \quad <?,Triangle,?,?,Yes>
Representing Version Spaces

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

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

Every member of the version space lies between these boundaries

$$VS_{H,D} = \{ h \in H \mid (\exists s \in S)(\exists g \in G)(g \geq h \geq s) \}$$

where $x \geq y$ means $x$ is more general or equal to $y$
Candidate Elimination Algorithm

\[ G = \text{maximally general hypotheses in } H \]
\[ S = \text{maximally specific hypotheses in } H \]

For each training example \( d \), do

If \( d \) is a positive example

Remove from \( G \) any hypothesis that does not include \( d \)
For each hypothesis \( s \) in \( S \) that does not include \( d \)

Remove \( s \) from \( S \)
Add to \( S \) all minimal generalizations \( h \) of \( s \) such that

1. \( h \) includes \( d \), and
2. Some member of \( G \) is more general than \( h \)
Remove from \( S \) any hypothesis that is more general than another hypothesis in \( S \)
Candidate Elimination Algorithm (cont)

For each training example $d$, do (cont)

If $d$ is a negative example

Remove from S any hypothesis that does include $d$

For each hypothesis $g$ in G that does include $d$

Remove $g$ from G

Add to G all minimal generalizations $h$ of $g$ such that

1. $h$ does not include $d$, and
2. Some member of S is more specific than $h$

Remove from G any hypothesis that is less general than another hypothesis in G

If G or S ever becomes empty, data not consistent (with H)
Example Trace

G1 G0: \{<?,?,?,?,?>\}

G2: \{<R,?,?,?,?>, <?,T,?,?,?>, <?,?,R,?,?>, <?,?,?,P,?>\}

G5 G4: \{<?,T,?,Y>, <?,?,R,?,Y>\}

S5: \{<?,?,R,?,Y>\}

S4 S3: \{<?,T,R,?,Y>\}

S2 S1: \{<R,T,R,P,Y>\}

S0: \{<Ø,Ø,Ø,Ø,Ø>\}

X2=<S,S,S,G,Y> -

X4=<R,T,R,G,N> -

X5=<S,S,R,Y,Y> +

X3=<S,T,R,Y,Y> +

X1=<R,T,R,P,Y> +
What Training Example Next?

G: \{ <?,?,Round,?,?>, <?,Triangle,?,?,?> \}

S: \{ <?,Triangle,Round,?,Yes> \}

<?,?,Round,?,Yes>  <?,Triangle,Round,?,?,>  <?,Triangle,?,?,Yes>
How Should These Be Classified?

G: \{ <?,?,Round,?,?> <?,Triangle,?,?,?> \}

S: \{ <?,Triangle,?,?,Yes> \}

<?,?,Round,?,Yes> <?,Triangle,?,?,?> <?,Triangle,?,?,Yes>
What Justifies this Inductive Leap?

+ < Round, Triangle, Round, Purple, Yes >
+ < Square, Triangle, Round, Yellow, Yes >

S: < ?, Triangle, Round, ?, Yes >

Why believe we can classify the unseen?

< Square, Triangle, Round, Purple, Yes > ?
An UN-Biased Learner

Idea: Choose $H$ that expresses every teachable concept (i.e., $H$ is the power set of $X$)

Consider $H' = \text{disjunctions, conjunctions, negations over previous } H$

For example:

$< ?, Triangle, Round, ?, Yes > \lor < Square, Square, ?, Purple, ? >$

What are $S, G$, in this case?
Inductive Bias

Consider

- concept learning algorithm $L$
- instances $X$, target concept $c$
- training examples $D_c = \{ <x, c(x)> \}$
- let $L(x_i, D_c)$ denote the classification assigned to the instance $x_i$ by $L$ after training on data $D_c$.

Definition:

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

$$(\forall x_i \in X)[(B \land D_c \land x_i) \models L(x_i, D_c)]$$

where $A \models B$ means $A$ logically entails $B$
Inductive Systems and Equivalent Deductive Systems

Inductive System

Training examples

Candidate Elimination Algorithm

Using Hypothesis Space H

New instance

Classification of new instance, or "don't know"

Equivalent Deductive System

Training examples

New instance

Assertion "H contains hypothesis"

Theorem Prover

Classification of new instance, or "don't know"
Three Learners with Different Biases

1. *Rote learner*: store examples, classify new instance iff it matches previously observed example (don’t know otherwise).

2. *Version space candidate elimination algorithm*.

3. *Find-S*
Summary Points

1. Concept learning as search through $H$
2. General-to-specific ordering over $H$
3. Version space candidate elimination algorithm
4. $S$ and $G$ boundaries characterize learner’s uncertainty
5. Learner can generate useful queries
6. Inductive leaps possible only if learner is biased
7. Inductive learners can be modeled by equivalent deductive systems