# MACHINE LEARNING: SYMBOL-BASED

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>9.0</td>
<td>Introduction</td>
</tr>
<tr>
<td>9.1</td>
<td>A Framework for Symbol-based Learning</td>
</tr>
<tr>
<td>9.2</td>
<td>Version Space Search</td>
</tr>
<tr>
<td>9.3</td>
<td>The ID3 Decision Tree Induction Algorithm</td>
</tr>
<tr>
<td>9.4</td>
<td>Inductive Bias and Learnability</td>
</tr>
<tr>
<td>9.5</td>
<td>Knowledge and Learning</td>
</tr>
<tr>
<td>9.6</td>
<td>Unsupervised Learning</td>
</tr>
<tr>
<td>9.7</td>
<td>Reinforcement Learning</td>
</tr>
<tr>
<td>9.8</td>
<td>Epilogue and References</td>
</tr>
<tr>
<td>9.9</td>
<td>Exercises</td>
</tr>
</tbody>
</table>

---

George F Luger

**ARTIFICIAL INTELLIGENCE**

Structure and Strategies for Complex Problem Solving

Fourth Edition
Figure 9.1: A general model of the learning process.
Figure 9.2: Examples and near misses for the concept “arch.”
Figure 9.3: Generalization of descriptions to include multiple examples.

a. An example of an arch and its network description

b. An example of another arch and its network description

(continued on next slide)
Figure 9.3: Generalization of descriptions to include multiple examples.

(continued from previous slide)

c. Given background knowledge that bricks and pyramids are both types of polygons

\[
\begin{align*}
\text{polygon} & \quad \text{isa} \quad \text{isa} \\
\text{brick} & \quad \text{pyramid}
\end{align*}
\]

d. Generalization that includes both examples

\[
\begin{align*}
\text{arch} & \quad \text{part} \quad \text{part} \quad \text{part} \\
\text{brick} & \quad \text{supports} \quad \text{supports} \quad \text{polygon}
\end{align*}
\]
a. Candidate description of an arch

b. A near miss and its description

c. Arch description specialized to exclude the near miss

Figure 9.4: Specialization of a description to exclude a near miss. In 9.4c we add constraints to 9.4a so that it can’t match with 9.4b.
Figure 9.5: A concept space.
Defining *specific to general search*, for hypothesis set \( S \) as:

**Begin**

*Initialize* \( S \) to the first positive training instance;
\( N \) is the set of all negative instances seen so far;

**For each positive instance** \( p \)

**Begin**

*For every* \( s \in S \), if \( s \) does not match \( p \), replace \( s \) with its most specific generalization that matches \( p \);

*Delete from* \( S \) all hypotheses more general than some other hypothesis in \( S \);

*Delete from* \( S \) all hypotheses that match a previously observed negative instance in \( N \);

**End**;

**For every negative instance** \( n \)

**Begin**

*Delete all members of* \( S \) that match \( n \);

*Add* \( n \) to \( N \) to check future hypotheses for overgeneralization;

**End**;

**End**
In this algorithm, negative instances lead to the specialization of candidate concepts; the algorithm uses positive instances to eliminate overly specialized concepts.

Begin
Initialize G to contain the most general concept in the space;
P contains all positive examples seen so far;

For each negative instance n
  Begin
  For each g ∈ G that matches n, replace g with its most general specializations that do not match n;
  Delete from G all hypotheses more specific than some other hypothesis in G;
  Delete from G all hypotheses that fail to match some positive example in P;
  End;

For each positive instance p
  Begin
  Delete from G all hypotheses that fail to match p;
  Add p to P;
  End;
End
Figure 9.6: The role of negative examples in preventing overgeneralization.

Concept induced from positive examples only

Concept induced from positive and negative examples
Figure 9.7: Specific to general search of the version space learning the concept “ball.”

\[
\begin{align*}
S & : \emptyset \\
\downarrow \\
S & : \{\text{obj}(\text{small, red, ball})\} \\
\downarrow \\
S & : \{\text{obj}(\text{small, X, ball})\} \\
\downarrow \\
S & : \{\text{obj}(Y, X, \text{ball})\}
\end{align*}
\]

Positive:

\[
\begin{align*}
\emptyset & : \text{obj}(\text{small, red, ball}) \\
\{\text{obj}(\text{small, red, ball})\} & : \text{obj}(\text{small, white, ball}) \\
\{\text{obj}(\text{small, X, ball})\} & : \text{obj}(\text{large, blue, ball}) \\
\{\text{obj}(Y, X, \text{ball})\} &
\end{align*}
\]
The algorithm specializes $G$ and generalizes $S$ until they converge on the target concept. The algorithm is defined:

Begin
Initialize $G$ to be the most general concept in the space;
Initialize $S$ to the first positive training instance;

For each new positive instance $p$
  Begin
  Delete all members of $G$ that fail to match $p$;
  For every $s \in S$, if $s$ does not match $p$, replace $s$ with its most specific generalizations that match $p$;
  Delete from $S$ any hypothesis more general than some other hypothesis in $S$;
  Delete from $S$ any hypothesis more general than some hypothesis in $G$;
  End;

For each new negative instance $n$
  Begin
  Delete all members of $S$ that match $n$;
  For each $g \in G$ that matches $n$, replace $g$ with its most general specializations that do not match $n$;
  Delete from $G$ any hypothesis more specific than some other hypothesis in $G$;
  Delete from $G$ any hypothesis more specific than some hypothesis in $S$;
  End;
Figure 9.8: General to specific search of the version space learning the concept “ball.”

\[ G: \{ \text{obj}(X,Y,Z) \} \]

\[ \text{Negative: obj(small, red, brick)} \]

\[ G: \{ \text{obj}(\text{large}, Y, Z), \text{obj}(X, \text{white}, Z), \text{obj}(X, \text{blue}, Z), \text{obj}(X, Y, \text{ball}), \text{obj}(X, Y, \text{cube}) \} \]

\[ \text{Positive: obj(\text{large, white, ball})} \]

\[ G: \{ \text{obj}(\text{large}, Y, Z), \text{obj}(X, \text{white}, Z), \text{obj}(X, Y, \text{ball}) \} \]

\[ \text{Negative: obj(\text{large, blue, cube})} \]

\[ G: \{ \text{obj}(\text{large, white, Z}), \text{obj}(X, \text{white, Z}), \text{obj}(X, Y, \text{ball}) \} \]

\[ \text{Positive: obj(\text{small, blue, ball})} \]

\[ G: \{ \text{obj}(X, Y, \text{ball}) \} \]
Figure 9.9: The candidate elimination algorithm learning the concept “red ball.”

- **G**: \{obj(X, Y, Z)\}
  - **S**: {}  
  - **Positive**: obj(small, red, ball)

- **G**: \{obj(X, Y, Z)\}
  - **S**: \{obj(small, red, ball)\}
  - **Negative**: obj(small, blue, ball)

- **G**: \{obj(X, red, Z)\}
  - **S**: \{obj(small, red, ball)\}
  - **Positive**: obj(large, red, ball)

- **G**: \{obj(X, red, Z)\}
  - **S**: \{obj(X, red, ball)\}
  - **Negative**: obj(large, red, cube)

- **G**: \{obj(X, red, ball)\}
  - **S**: \{obj(X, red, ball)\}
Figure 9.10: Converging boundaries of the G and S sets in the candidate elimination algorithm.
Figure 9.11: A portion of LEX’s hierarchy of symbols.
Figure 9.12: A version space for OP2, adapted from Mitchell et al. (1983).

\[
G: \int f_1(x) f_2(2) \, dx \rightarrow \text{apply OP2}
\]

\[
\int \text{poly}(x) f_2(x) \, dx \rightarrow \text{apply OP2}
\]

\[
\int \text{transc}(x) f_2(x) \, dx \rightarrow \text{apply OP2}
\]

\[
\int kx \cos(x) \, dx \rightarrow \text{apply OP2}
\]

\[
\int 3x \cos(x) \, dx \rightarrow \text{apply OP2}
\]

\[
\int 3x \sin(x) \, dx \rightarrow \text{apply OP2}
\]