Machine Learning Techniques

UNIT - III

1.1. Well Posed Learning Problems:

Learning is broadly defined to include any computer program that improves its performance at some task through experience.

"A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P. if its performance at tasks in T, as measured by P. improves with experience E."

In general, to have a well-defined learning problem, three features including

ⓐ the class of tasks
ⓑ the measure of performance to be improved.
ⓒ the source of experience.

must be identified.

Ex1: A checkers learning problem:

- Task T: Playing checkers.
- Performance measure P: percent of games won against opponents.
- Training Experience E: Playing practice games against itself.

Ex2 : A handwritten text recognition learning pr

- T: recognizing and classifying handwritten w within images
- P: percent of words correctly classified.
- E: a database of handwritten words with given classifications.

Ex3: A robot driving learning problem:

- T: Driving on public four-lane highways using vision sensors.
- P: average distance traveled before an error (as judged by human overseer).
- E: a sequence of images and steering commands recorded while observing a human driver.

1.2. Designing a Learning System:

In order to illustrate basic design issues and approaches to ML, Let us consider learning checkers game, with the goal of entering it in the world of checkers tournament.

1.2.1. Choosing the Training Experience:-

The first design choice faced is to choose the type of experience from which the system has to learn.

The type of experience available will have significant impact on success/failure of learner.

The key attribute of learning is that whether the training experience provides direct/indirect feedback regarding the choice made by the learning system.

Direct : individual board state and its correct move.

Indirect : move sequence and final outcomes of various games played.

The second attribute of the training experience is the degree to which the learner controls the sequence of training examples.

The third import attribute of traing experience is how well it represents the distribution of examples over which the final system performance P must be measured.

In order to complete the design of learning system we must choose

1. the exact type of knowledge to be learned.
2. a representation for this target knowledge.
3. a learning mechanism.

1.2.2. Choosing the Target Function.

The next design choice is to determine exactly what type of knowledge will be learned and how this will be used by the performance program.

In checkers game, there will be a set of legal moves from each board state. It is the task of system to choose best move. However a strategy to choose the best move is not known.

It is the task of system to learn to choose a best move from available legal moves. This function can be considered as chooseMove.

⇒ ChooseMove : B → M.

This function accepts B (any board) and produce output as M.

where B : set of legal board states.

M : set of legal moves

ChooseMove is a target function which is an important part of design choice.

Implementation of target function will be very difficult in case of indirect experience.

Thus, the alternative target function is an evaluation function that assigns a

numerical score to any given board state
Let this target function be V.

$V : B \rightarrow R$.

V maps any legal board from B to some real value $\in R$.

If system able to learn such target function then it can easily select the best move from each board $\in B$.

The value of each board can be decided based on problem addressed.

Ex:-

1. If b is a final board that is won then $V(b) = 100$.
2. If b is a final board that is lost then $V(b) = -100$
3. If b is a final board that is drawn, $V(b) = 0$.
4. If b is not a final board, then $V(b) = V(b')$ where b' is the best final board state that can be achieved starting from b and playing optimally until the end of the game.

This is non-operational ~~operational~~ solution as in case of 4, determining the value of $V(b)$ is tedious which requires search ahead. It is difficult to define the function perfectly hence we consider approximation function represented as $\hat{V}$.

1.2.3. Choosing a Representation for the Target Fu

once an ideal target function V is specified, then must choose a representation that the learning pro -gram will use to describe the function $\hat{V}$ that it lear

There are many choices like:

1. Allow the program to represent $\hat{V}$ using a large table with a distinct entry specifying the value for each distinct board state.
2. Allow the program to represent $\hat{V}$ using a collection of rules that match against features of the board state

Ex representation :-

Let x_1 : no of black pieces on board
x_2 : no of red pieces on board
x_3 : no of black kings on board
x_4 : no of red kings on board.
x_5 : no of black threatened by red
x_6 : no of red threatened by black.

Then the learning program represent $\hat{V}(b)$ as a linear function of the form.

$$\hat{V}(b) = w_0 + w_1 x_1 + w_2 x_2 + w_3 x_3 + w_4 x_4 + w_5 x_5 + w_6 x_6$$

where $w_0 - w_6$ are numerical coefficients or weights, to be chosen by the learning algorithm. The learned values of $w_1 - w_6$ determine the relative importance of the various board features in determining the value of the board, w_0 will provide an additive constant to the board value.

Choosing a function approximation algorithm.

In order to learn the target fun $\hat{V}$, it requires a set of training examples, each describing a specific board state b and the training value $V_{train}(b)$ for b.

i) The training example is an ordered pair of the form $\langle b, V_{train}(b) \rangle$

Ex:- $\langle \langle x_1=3, x_2=0, x_3=1, x_4=0, x_5=0, x_6=0 \rangle, +100 \rangle$

2 procedures that derives such training examples from the indirect training experience available to the learner, then adjust the weights w_i to best fit these training example are as follows.

1.2.4.1. Estimating Training values:

There is an ambiguity in assigning the score to each intermediate board.

Dispite the ambiguity inherent in estimating training values for intermediate board states, one simple approach has been found to be surprisingly successful.

i) assign the training value of $V_{train}(b)$ for any intermediate board state b to be $\hat{V}(Successor(b))$.

where $\hat{V}$ is the learner's current approximation to V and where Successor(b) denotes the next board state following b for which it is again the program's turn to move.

Rule for estimating traing values.

$$V_{train}(b) \leftarrow \hat{V}(Successor(b))$$

1.2.4.2. Adjusting the weights

A common approach to best fit the training data is to minimize the squared error between the training values and the predicted values through hypothesis $\hat{V}$.

$$E = \sum_{\langle b, V_{train}(b) \rangle \in \text{training example}} (V_{train}(b) - \hat{V}(b))^2$$

LMS weight update rule.

For each training example $\langle b, V_{train}(b) \rangle$

- Use the current wt to calculate $\hat{V}(b)$
- For each wt w_i, update it as

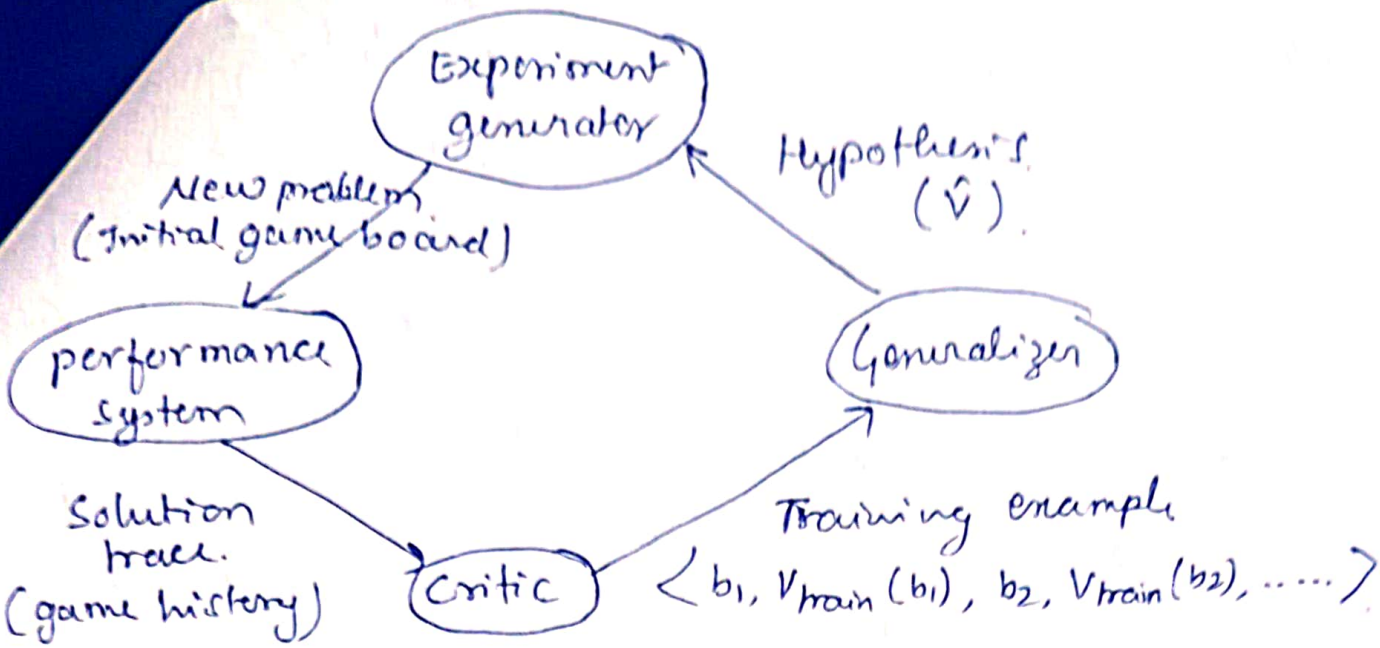
$$w_i \leftarrow w_i + \eta (V_{train}(b) - \hat{V}(b)) x_i$$

where η is a small constant that moderates the size of the wt update

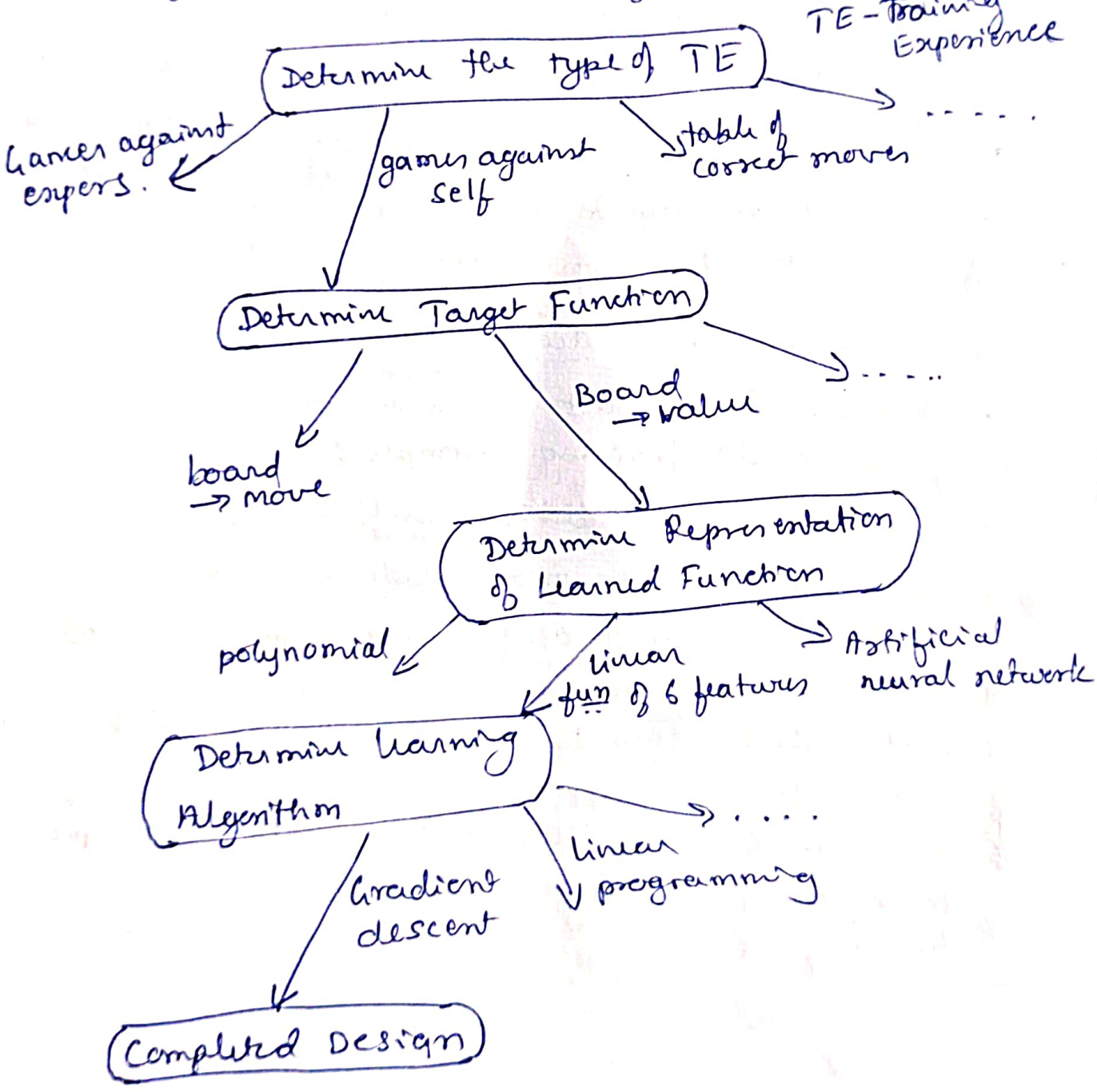
1.2.5 The final design

Finally the checkers learning system can be described by 4 distinct modules that represent the central components in many learning systems.

(a) Performance system.
(b) Critic.
(c) Generalizer.
(d) Experiment generator.



Summary of choice in designing checkers problem.



1.3. Perspectives and Issues in Machine Learning

One useful perspective on ML is that it involves searching a very large space of possible hypotheses to determine one that best fits the observed data and any prior knowledge held by the learner.

Thus the learners task is to search through the vast space of hypothesis to locate the hypothesis that is most consistent with the available training example.

Least mean square (LMS) algorithm for fitting weights achieves this goal by iteratively tuning the weights, adding a correction to each weight each time the hypothesized evaluation function predicts a value that differ from the training value.

1.3.1. Issues in Machine Learning

1. what algorithms exist for learning general target fun from specific training examples?
2. How much training data is sufficient?
3. When and how can prior knowledge held by the learner guide the process of generalizing from examples?
4. what is the best strategy for choosing a useful next TE?
5. what is the best way to reduce the learning task to one or more function approximation problem?
6. How can the learner automatically alter its representation to improve its ability to represent and learn the target function.

A concept learning task

Let following table represents training example

Ex	Sky	AirTemp	Humidity	wind	Water	Forecast	EnjoySport
1	Sunny	Warm	Normal	Strong	warm	same	yes
2	Sunny	Warm	High	Strong	warm	same	yes
3.	Rainy	Cold	High	Strong	warm	change	No.
4.	Sunny	Warm	High	Strong	Cool	change	yes.

EnjoySport is the target attribute whereas remaining six attributes are independent variables, which influence the target attribute.

Task here is to identify the days on which Mr. Aldo enjoy the water sport.

Thus the learner has to identify the best hypothesis to map with all instances consistently.

Here for this example training data each hypothesis be a vector of six constraints, specifying the values of the Sky, AirTemp, Humidity, wind, water, and Forcast. For each attribute the hypothesis will either

- indicate by a "?" that any value is acceptable for this attribute
- Specify a single required value (Ex: warm) for the attribute
- Indicate by a '0' that no value is acceptable

If some instance x satisfies all the constraints hypothesis h, then h classifies x as a +ve example ($h(x) = 1$)

Ex:- <?, cold, High, ?, ?, ?>

This hypothesis implies that the Aldo enjoys his favourite sport only on cold days with high humidity and independent of other attributes

The most general hypothesis that every day is a +ve example is represented as <?, ?, ?, ?, ?, ?>

and the most specific possible hypothesis that no day is a +ve example is represented as <∅, ∅, ∅, ∅, ∅, ∅>.

Finally it can be summarized that the EnjoySport concept learning task requires learning the set of days for which EnjoySport = Yes, describing this set by a conjunction of constraints over the instance attributes.

In general, any concept learning task can be described by the set of instances over which the target function is defined. i.e. the target function is the set of candidate hypotheses considered by the learner, and the set of available training examples.

2.2.1. Notation

x -> The set of items over which the concept is defined is called set of instances

In current example the x is a set of all days each representing six attributes.

c → The concept or function to be learned is called the target concept

Generally, 'c' can be any boolean valued function defined over the instances x.

ie $c : X \rightarrow \{0, 1\}$

Ex: $c(x) = 1$ if EnjoySport = Yes.

$c(x) = 0$ if EnjoySport = NO.

Thus the ordered pair is re-represented as $\langle x, c(x) \rangle$. where x is an instance $\in X$ and $c(x)$ is a target value in X.

2.2.2. 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.

2.3. Concept learning as search

The concept learning can be viewed as a task of Searching through a large space of data to identify hypothesis that best fits the Training Example(TE). The goal of search is to find best hypothesis.

Let X be the instances

H be the hypothesis

in EnjoySport Learning ~~Space~~ task.

Given that the attribute Sky has 3 possible values and all remaining 5 attributes, each have 2 possible values. the instance space X contains exactly

$3 \cdot 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 = 96$ distinct instances

lly there are $5 \cdot 4 \cdot 4 \cdot 4 \cdot 4 \cdot 4 = 5120$ Syntactically
hypothesis within H.

However every hypothesis containing one or more Symbols represents the empty set of instances; that is it classifies every instance as negative.

Hence the no of symantically distinct hypothesis is only $1 + (4 \cdot 3 \cdot 3 \cdot 3 \cdot 3 \cdot 3) = 973$.

An the EnjoySport is simple learning task, the hypothesis space is relatively smaller. But in most practical learning tasks, the hypothesis spaces will be much more larger, sometimes infinite as well.

2.3.1. Generic-to-Specific ordering of Hypothesis

Many algorithms for concept learning organize the search through the hypothesis space by relying on a very useful structure that exists for any concept learning problem:

ii:- a generic-to-specific ordering of hypothesis

By taking the advantage of this naturally occuring structure over the hypothesis space, we can design learning algorithms that exhaustively search even infinite hypothesis spaces without explicitely enumerating every hypothesis.

Let us consider two hypothesis for illustration.

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

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

As h_2 imposes fewer constraints on the instances, it classifies more instances as +ve.

Also, any instance classified +ve by h_1 will also be classified +ve by h_2.

Thus, h_2 is more general than h_1.

With this example "more general than" relationship between hypotheses can be defined as follows:

For any instance $x \in X$ and the hypothesis $h \in H$.

we say that x satisfies h iff $h(x) = 1$.

Now let us define the "more-general-than-or-equal-to" relation in terms of the sets of instances that satisfy the 2 hypotheses:

Given hypotheses h_j and h_k, then h_j is more-general-than-or-equal-to h_k iff any instance that satisfies h_k also satisfies h_j.

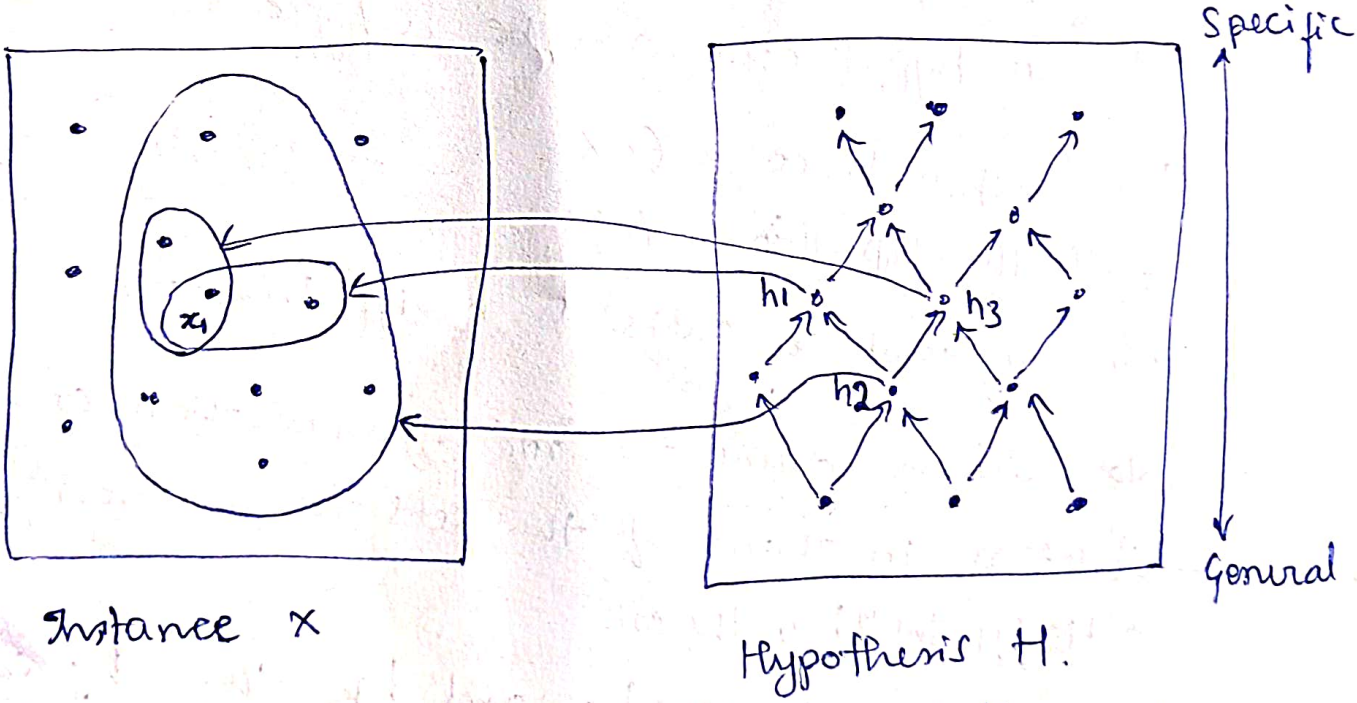
Definition:

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 ($h_j \geq_g h_k$) iff

$$(\forall x \in X)[(h_k(x) = 1) \rightarrow (h_j(x) = 1)]$$

However one hypothesis can be strictly more-general than the other. h_j is (strictly) more-general-than h_k ($h_j >_g h_k$) iff $(h_j \geq_g h_k) \wedge (h_k \not\geq_g h_j)$.

Finally, we will sometimes find the inverse useful and will say that h_j is more-specific-than h_k when h_k is more-general-than h_j.



Though there is a intersection among the instances accepted by h_1 and h_3, still h_1 is not more-general than h_3 or vice versa. However h_2 is more-general than both h_1 and h_3.

Find-S: Finding a Maximally Specific Hypothesis

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 is satisfied by x
     Then do nothing
     Else replace a_i in h by the next more general constraint that is satisfied by x.
3. output hypothesis h.

Illustration :-

consider the x in table of concept learning task.

1. Initialize hypothesis h to most specific hypo in H.

   $h = \langle \phi, \phi, \phi, \phi, \phi, \phi \rangle$.

2. Upon observing first instance x from table, the h is not fit over x. Hence replace each constraint in h with the next general constraint

   ie $h = \langle$Sunny, Warm, Normal, Strong, Warm, Same$\rangle$

3. Now consider next +ve x from x. we will have the hypothesis as

   $h = \langle$Sunny, warm, ?, Strong, warm, Same$\rangle$

4. Now consider next +ve x from X. and the constraints on

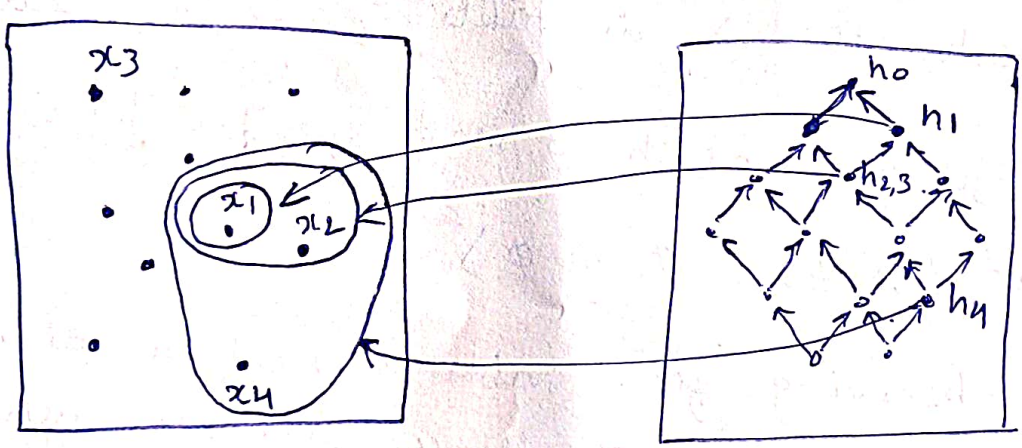
$h = \langle Sunny, warm, ?, Strong, ?, ? \rangle$.

5. The output hypothesis is

$h = \langle Sunny, warm, ?, Strong, ?, ? \rangle$

which consistently fit over the training examples considered for illustration of FIND-S algorithm.

The final hypothesis h is able to classify all four training examples correctly.



Instance Space X

Hypothesis Space H.

Here in diagram the $h_{2,3}$ is more-general-than h_1 and $h_{2,3}$.

Though the final hypothesis $h \in H$ is able to classify all the training example correctly (including -ve examples), still there are several questions left unanswered by this learning algorithm:

① Has the learner converged to the correct target concept?

There is no way to determine whether this is the only hypothesis consistent with data or are there other hypothesis.

② Why to prefer the most specific hypothesis? when there are so many hyp in H, it is unclear that why it starts with the most specific only.

③ Are the training example consistent? In practical example data, there is a chance of errors called noise. Such training example surely mislead find-S.

④ What if there are several maximally specific consistent hypotheses?

Solved Examples

(01)

TimeOfDay	weather	Temp	Company	Hum	Wind	
M	Sunny	warm	Yes	Mild	Strong	
E	Rainy	cold	No	Mild	Normal	No
M	Sunny	Moderate	Yes	Normal	Normal	Yes
E	Sunny	cold	Yes	High	Strong	Yes

Target to be identified is whether the person play Tennis or not. ie "Goes" is the target variable.

Soln:- $h_0 = \langle \phi, \phi, \phi, \phi, \phi, \phi \rangle$

$h_1 = \langle M, \text{sunny}, \text{warm}, \text{yes}, \text{Mild}, \text{strong} \rangle$

$h_2 = \langle M, \text{sunny}, ?, \text{yes}, ?, ? \rangle$

$h_3 = \langle ?, \text{sunny}, ?, \text{yes}, ?, ? \rangle$

He is the final and most specific hypothesis which is consistent over all four instances.

(2) Mammals classification is the target

Instance	Toothed	legs	Species.
1	True	True	mammal
2	True	False	Reptile
3	False	True	mammal
4	False	True	Reptile

Mammals classification.

Instance	BodyTemp	GivesBirth	FourLegged	Hibernates	Class label
1	warm	yes	yes	yes	yes
2	warm	yes	yes	No	yes
3	warm	No	No	No	No
4	cold	No	No	No	No
5	cold	yes	No	No	No
6	warm	yes	No	yes	yes
7.	warm	yes	No	No	Yes.

2.5. Version space and the candidate-elimination Algorith

Although Find-s outputs a hypothesis from H that is consistent with the training examples, this is just one of many hypotheses from H that might fit the training data equally well.

But, the key idea in candidate elimination algorithm is to output a description of the set of all hypotheses consistent with the training examples.

This algorithm computes the description of this ~~data~~ set without explicitely enumerating all of its members. and it is accomplished using more-general-than ordering

candidate elimination algorithm has been a...
problems such as learning regularities in ch...
mass spectroscopy and learning control rules
heuristic search.

Limitations of Find-S and candidate elimination algorith...

① they perform poorly in case of noisy training data.

2.5.1. Representation

Definitions :-

① First, let us say that a hypothesis is consistent with the training example (TE) if it correctly classifies all the examples.

∵ A hyp h is consistent with a set of TE D iff $h(x) = c(x)$ for each example $\langle x, c(x) \rangle$ in D.

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

② The Candidate-Elimination Algorithm (CEA) represents the set of all hypotheses consistent with observed TE. This subset of all hypothesis is called the version space w.r.t hypotheses space H and the TE D. because it contains all plausible versions of the target concept.

∵ The version space, denoted $VS_{H,D}$ w.r.t. hypothesis space H and T.E. D, is the subset of hypotheses.

...on H consistent with the TE in D

$$VS_{H,D} \equiv \{ h \in H \mid \text{consistent}(h, D) \}$$

2.5.2. List-then-eliminate algorithm

It first initializes the version space to contain all hypotheses in H, then eliminates any hypothesis found inconsistent with any TE. Thus the VS of hypothesis shrinks as more examples are observed until ~~an~~ ideally just one hypothesis remains that is consistent with all observed examples

Disadvantages:

① If insufficient data is given, then, it outputs entire set of hyp that is consistent with the data observed But that may not be the expected hyp for that particular problem/learning.

② It is application for only finite data set.

③ It needs enumeration of each hyp in H.

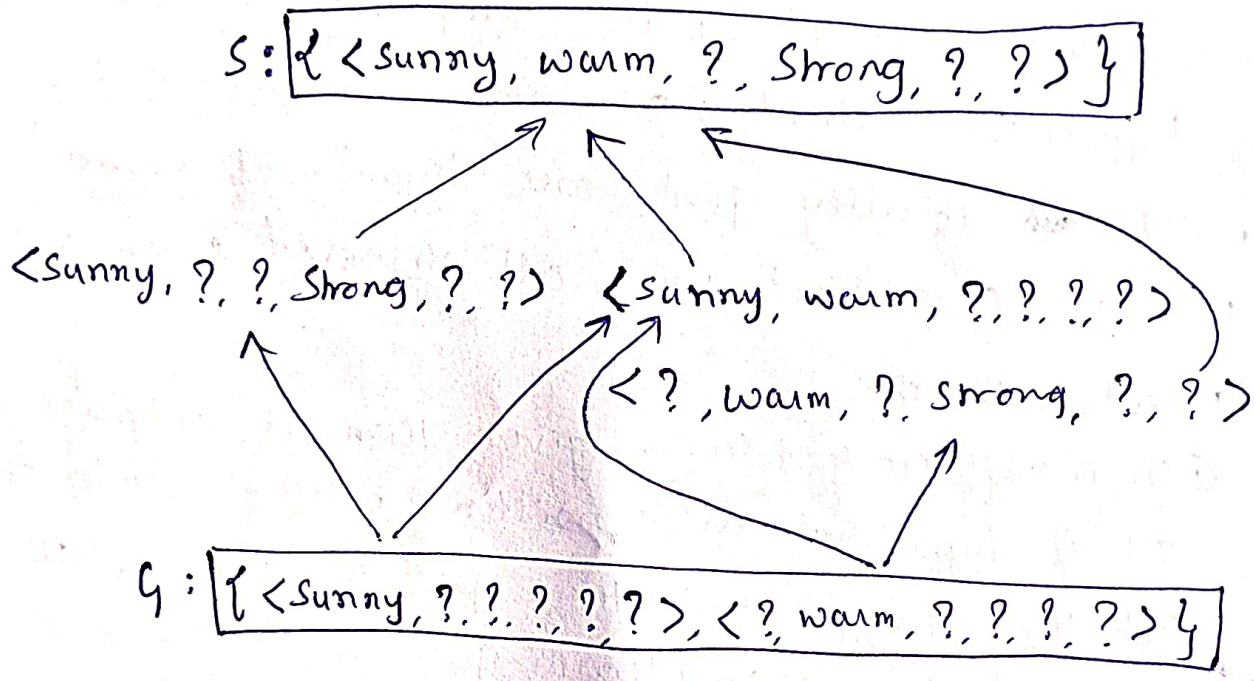
Advantage: It guarantees the consistent hyp

Algorithm: List-then-eliminate

1. Version space ← a list containing every hyp in H.
2. For each training example, $\langle x, c(x) \rangle$
   remove from VersionSpace any hyp h for which $h(x) \neq c(x)$.
3. Output the list of hyp in VersionSpace.

2.5.3. A more compact representation for Version

The version space is represented by its most and least-general members. These members form general and specific boundary sets that delimit the version space within the partially ordered hypothesis space.



2.5.4. Candidate elimination learning algorithm:

- Initialize G to set of maximally general hyp in H.
- Initialize S to set of maximally specific hyp in H.
- For each $d \in D$. do
  - *If d is +ve TE
    - → Remove from G any hyp inconsistent with d
    - → For each hyp 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 members of G is more general than h.
    - → Remove from S any hyp that is more general than another hyp in S

If d is -ve example

→ Remove from S any hyp that is inconsistent with d.

→ For each hyp 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 members of S is more specific than h.

→ Remove from G any hyp that is less general than another hyp in G.

Illustration ..

S_0 : {<ϕ ϕ ϕ ϕ ϕ ϕ>}.

S_1 : {<sunny, warm, Normal, Strong, warm, same>}

S_3 . S_2 : {<Sunny, warm, ?, Strong, warm, same>}.

S_4 : {<Sunny, warm, ?, Strong, ?, ?>}.

<Sunny, warm, ? ? ? ? >, <? warm, ?, Strong, ? ? >, <Sunny, warm, ? ? ? ?>}.

G_4 : {<Sunny, ? ? ? ? ? >, <?, warm, ? ? ? ? >}.

G_3 : {<Sunny, ? ? ? ? ? >, <?, warm, ? ? ? ? >, <? ? ? ? ? same>}.

G_2 ; G_1 ; G_0 : {<[illegible] ? ? ? ? ? ? >}.

e	of Day	Weather	Temp	Company	Humidity	Wind	Goes
		Sunny	Warm	Yes	Mild	Strong	Yes
E		Rainy	Cold	NO	Mild	Normal	NO
M		Sunny	Moderate	Yes	Normal	Normal	Yes
E		Sunny	Cold	Yes	High	strong	Yes

Mammals classification

2)

Instance	Toothed	Legs	Species
1	True	True	Mammal
2	True	False	Reptile
3	False	True	Mammal
4	False	True	Reptile.

Mammals classification

3)

Instance	Body Temp	Gives Birth	Four Legged	Hibernate	Class Label
1	Warm	Yes	Yes	Yes	Yes
2	Warm	Yes	Yes	No	Yes
3	Warm	No	No	No	No
4	Cold	No	No	No	No
5	Cold	Yes	No	No	No
6	Warm	Yes	No yes.	Yes	Yes
7	Warm	Yes	No	No	Yes

	Size	colour	shop	clabls.
1	Big	Red	circle	No
2	Small	Red	Triangle	No.
3	Small	Red	circle	Yes
4	Big	Blue	circle	No.
5	Small	Blue	circle	Yes.

	Eyes	Nose	Head	Fcolor	Hair	small
1	Round	Triangle	Round	Purple	Yes	Yes
2	square	Square	square	Green	Yes	No
3	Square	Triangle	Round	Yellow	Yes	Yes
4	Round	Triangle	Round	Green	No	No
5	Square	Square	Round	Yellow	Yes	Yes.