Lecture 1

Concept Learning
and the Version Space Algorithm

Thursday, August 26, 1999

William H. Hsu
Department of Computing and Information Sciences, KSU
http://www.cis.ksu.edu/~bhsu

Readings:
Chapter 2, Mitchell
Section 5.1.2, Buchanan and Wilkins

Lecture Outline

- **Read: Chapter 2, Mitchell; Section 5.1.2, Buchanan and Wilkins**
- **Suggested Exercises: 2.2, 2.3, 2.4, 2.6**
- **Taxonomy of Learning Systems**
- **Learning from Examples**
  - **(Supervised) concept learning framework**
  - **Simple approach: assumes no noise; illustrates key concepts**
- **General-to-Specific Ordering over Hypotheses**
  - **Version space: partially-ordered set (poset) formalism**
  - **Candidate elimination algorithm**
  - **Inductive learning**
- **Choosing New Examples**
- **Next Week**
  - **The need for inductive bias: 2.7, Mitchell; 2.4.1-2.4.3, Shavlik and Dietterich**
  - **Computational learning theory (COLT): Chapter 7, Mitchell**
  - **PAC learning formalism: 7.2-7.4, Mitchell; 2.4.2, Shavlik and Dietterich**

What to Learn?

- **Classification Functions**
  - **Learning hidden functions: estimating ("fitting") parameters**
  - **Concept learning (e.g., chair, face, game)**
  - **Diagnosis, prognosis: medical, risk assessment, fraud, mechanical systems**
- **Models**
  - **Map (for navigation)**
  - **Distribution (query answering, *aka* QA)**
  - **Language model (e.g., automaton/grammar)**
- **Skills**
  - **Playing games**
  - **Planning**
  - **Reasoning (acquiring representation to use in reasoning)**
- **Cluster Definitions for Pattern Recognition**
  - **Shapes of objects**
  - **Functional or taxonomic definition**
- ***Many Problems Can Be Reduced to Classification***

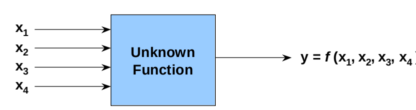
How to Learn It?

- **Supervised**
  - **What is learned? Classification function; other models**
  - **Inputs and outputs? Learning:** examples $\langle x, f(x) \rangle \rightarrow$ approximation $\hat{f}(x)$
  - **How is it learned? Presentation of examples to learner (by teacher)**
- **Unsupervised**
  - **Cluster definition, or *vector quantization* function (*codebook*)**
  - **Learning:** observations $x \times$ distance metric $d(x_1, x_2) \rightarrow$ discrete codebook $f(x)$
  - **Formation, segmentation, labeling of clusters based on observations, metric**
- **Reinforcement**
  - **Control policy (function from states of the world to actions)**
  - **Learning:** state/reward sequence $\{\langle s_i, r_i \rangle : 1 \leq i \leq n\} \rightarrow$ policy $p : s \rightarrow a$
  - **(Delayed) feedback of reward values to agent based on actions selected; model updated based on reward, (partially) observable state**

(Supervised) Concept Learning

- **Given: Training Examples <*x, f(x)*> of Some Unknown Function *f***
- **Find: A Good Approximation to *f***
- **Examples (besides Concept Learning)**
  - **Disease diagnosis**
    - ***x* = properties of patient (medical history, symptoms, lab tests)**
    - ***f* = disease (or recommended therapy)**
  - **Risk assessment**
    - ***x* = properties of consumer, policyholder (demographics, accident history)**
    - ***f* = risk level (expected cost)**
  - **Automatic steering**
    - ***x* = bitmap picture of road surface in front of vehicle**
    - ***f* = degrees to turn the steering wheel**
  - **Part-of-speech tagging**
  - **Fraud/intrusion detection**
  - **Web log analysis**
  - **Multisensor integration and prediction**

A Learning Problem



Example	x_1	x_2	x_3	x_4	y
0	0	1	1	0	0
1	0	0	0	0	0
2	0	0	1	1	1
3	1	0	0	1	1
4	0	1	1	0	0
5	1	1	0	0	0
6	0	1	0	1	0

- **x_i: t_i, y: t, *f*: $(t_1 \times t_2 \times t_3 \times t_4) \rightarrow t$**
- **Our learning function: Vector $(t_1 \times t_2 \times t_3 \times t_4 \times t) \rightarrow (t_1 \times t_2 \times t_3 \times t_4) \rightarrow t$**

Hypothesis Space: Unrestricted Case

- $| \mathcal{A} \rightarrow \mathcal{B} | = | \mathcal{B} |^{|\mathcal{A}|}$
- $|\mathcal{H}^4 \rightarrow \mathcal{H}| = | \{0,1\} \times \{0,1\} \times \{0,1\} \times \{0,1\} \rightarrow \{0,1\} | = 2^{2^4} = 65536$ function values
- **Complete Ignorance: Is Learning Possible?**
  - Need to see every possible input/output pair
  - After 7 examples, still have $2^9 = 512$ possibilities (out of 65536) for *f*

Example	x_1	x_2	x_3	x_4	y
0	0	0	0	0	?
1	0	0	0	1	?
2	0	0	1	0	0
3	0	0	1	1	1
4	0	1	0	0	0
5	0	1	0	1	0
6	0	1	1	0	0
7	0	1	1	1	?
8	1	0	0	0	?
9	1	0	0	1	1
10	1	0	1	0	?
11	1	0	1	1	?
12	1	1	0	0	0
13	1	1	0	1	?
14	1	1	1	0	?
15	1	1	1	1	?

Training Examples for Concept *EnjoySport*

- **Specification for Examples**
  - Similar to a data type definition
  - 6 attributes: Sky, Temp, Humidity, Wind, Water, Forecast
  - Nominal-valued (symbolic) attributes - enumerative data type
- **Binary (Boolean-Valued or $\mathcal{H}$-Valued) Concept**
- **Supervised Learning Problem: *Describe the General Concept***

Example	Sky	Air Temp	Humidity	Wind	Water	Forecast	Enjoy Sport
0	Sunny	Warm	Normal	Strong	Warm	Same	Yes
1	Sunny	Warm	High	Strong	Warm	Same	Yes
2	Rainy	Cold	High	Strong	Warm	Change	No
3	Sunny	Warm	High	Strong	Cool	Change	Yes

Representing Hypotheses

- **Many Possible Representations**
- **Hypothesis *h*: Conjunction of Constraints on Attributes**
- **Constraint Values**
  - Specific value (e.g., *Water = Warm*)
  - Don't care (e.g., *"Water = ?"*)
  - No value allowed (e.g., "Water = Ø")
- **Example Hypothesis for *EnjoySport***
  - Sky AirTemp Humidity Wind Water Forecast
    <Sunny ? ? Strong ? Same>
  - Is this consistent with the training examples?
  - What are some hypotheses that are consistent with the examples?

Prototypical Concept Learning Tasks

- **Given**
  - Instances X: possible days, each described by attributes *Sky, AirTemp, Humidity, Wind, Water, Forecast*
  - Target function *c* ≡ EnjoySport: $X \rightarrow \mathcal{H} \equiv$ {{Rainy, Sunny} × {Warm, Cold} × {Normal, High} × {None, Mild, Strong} × {Cool, Warm} × {Same, Change}} → {0, 1}
  - Hypotheses *H*: conjunctions of literals (e.g., *<?, Cold, High, ?, ?, ?>*)
  - Training examples *D*: positive and negative examples of the target function

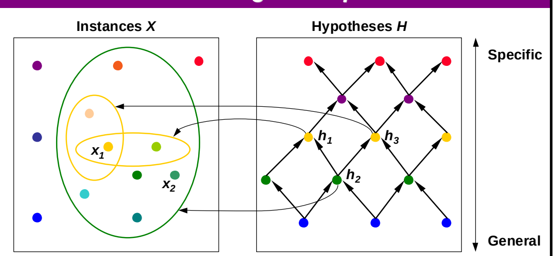
$$\langle x_1, c(x_1) \rangle, \ldots, \langle x_m, c(x_m) \rangle$$

- **Determine**
  - Hypothesis $h \in H$ such that $h(x) = c(x)$ for all $x \in D$
  - Such *h* are consistent with the training data
- **Training Examples**
  - Assumption: no missing *X* values
  - Noise in values of *c* (contradictory labels)?

The Inductive Learning Hypothesis

- **Fundamental Assumption of Inductive Learning**
- **Informal Statement**
  - 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
  - Definitions deferred: *sufficiently large, approximate well, unobserved*
- **Later: Formal Statements, Justification, Analysis**
  - Chapters 5-7, Mitchell: different operational definitions
  - Chapter 5: statistical
  - Chapter 6: probabilistic
  - Chapter 7: computational
- **Next: How to *Find* This Hypothesis?**

Instances, Hypotheses, and the Partial Ordering *Less-Specific-Than*



x_1 = *<Sunny, Warm, High, Strong, Cool, Same>*
x_2 = *<Sunny, Warm, High, Light, Warm, Same>*

h_1 = *<Sunny, ?, ?, Strong, ?, ?>*
h_2 = *<Sunny, ?, ?, ?, ?, ?>*
h_3 = *<Sunny, ?, ?, ?, Cool, ?>*

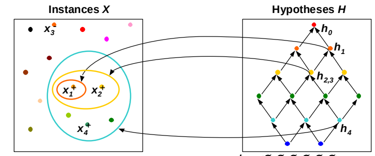
$\leq_P$ ≡ *Less-Specific-Than* ≡ *More-General-Than*

$h_2 \leq_P h_1$
$h_2 \leq_P h_3$

Find-S Algorithm

1. Initialize *h* to the most specific hypothesis in *H*
   - *H*: the *hypothesis space* (partially ordered set under relation *Less-Specific-Than*)
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 the next more general constraint that is satisfied by *x*
3. Output hypothesis *h*

Hypothesis Space Search by *Find-S*



x_1 = <Sunny, Warm, Normal, Strong, Warm, Same>, +
x_2 = <Sunny, Warm, High, Strong, Warm, Same>, +
x_3 = <Rainy, Cold, High, Strong, Warm, Change>, -
x_4 = <Sunny, Warm, High, Strong, Cool, Change>, +

h_1 = <Ø, Ø, Ø, Ø, Ø, Ø>
h_2 = <Sunny, Warm, Normal, Strong, Warm, Same>
h_3 = <Sunny, Warm, ?, Strong, Warm, Same>
h_4 = <Sunny, Warm, ?, Strong, Warm, Same>
h_5 = <Sunny, Warm, ?, Strong, ?, ?>

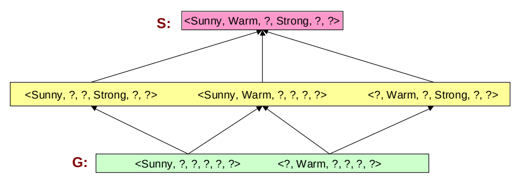
- Shortcomings of *Find-S*
  - Can't tell whether it has learned concept
  - Can't tell when training data inconsistent
  - Picks a maximally specific *h* (why?)
  - Depending on *H*, there might be several!

Version Spaces

- Definition: Consistent Hypotheses
  - 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 *<x, c(x)>* in *D*.
  - *Consistent* (*h*, *D*) $\equiv \forall$ <x, c(x)> $\in$ *D* . *h(x) = c(x)*
- Definition: Version Space
  - 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*.
  - $VS_{H,D} \equiv \{ h \in H \mid Consistent\ (h, D) \}$

The *List-Then-Eliminate* Algorithm

1. Initialization: *VersionSpace* $\leftarrow$ 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*



Example Version Space

Representing Version Spaces

- Hypothesis Space
  - A finite *meet semilattice* (partial ordering Less-Specific-Than; $\perp \equiv$ all *?*)
  - Every pair of hypotheses has a *greatest lower bound* (GLB)
  - $VS_{H,D} \equiv$ the consistent poset (partially-ordered subset of *H*)
- Definition: General Boundary
  - General boundary G of version space $VS_{H,D}$: set of most general members
  - Most general $\equiv$ *minimal* elements of $VS_{H,D} \equiv$ "set of necessary conditions"
- Definition: Specific Boundary
  - Specific boundary S of version space $VS_{H,D}$: set of most specific members
  - Most specific $\equiv$ *maximal* elements of $VS_{H,D} \equiv$ "set of sufficient conditions"
- Version Space
  - Every member of the version space lies between *S* and *G*
  - $VS_{H,D} \equiv \{ h \in H \mid \exists\, s \in S . \exists\, g \in G . g \lesssim_P h \lesssim_P s \}$ where $\lesssim_P \equiv$ Less-Specific-Than

Candidate Elimination Algorithm [1]

1. Initialization
   - *G* $\leftarrow$ (singleton) set containing most general hypothesis in *H*, denoted {<?, … , ?>}
   - S $\leftarrow$ set of most specific hypotheses in *H*, denoted {<Ø, … , Ø>}
2. For each training example *d*
   - If *d* is a positive example (*Update-S*)
     - Remove from G any hypotheses 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* such that
         1. *h* is consistent with *d*
         2. Some member of *G* is more general than *h*
       - (These are the greatest lower bounds, or *meets*, $s \vee d$, in $VS_{H,D}$)
       - Remove from *S* any hypothesis that is more general than another hypothesis in *S* (remove any dominated elements)

Candidate Elimination Algorithm [2]

(continued)

If *d* is a negative example (*Update-G*)

Remove from *S* any hypotheses inconsistent with *d*

For each hypothesis *g* in *G* that is not consistent with *d*

Remove *g* from *G*

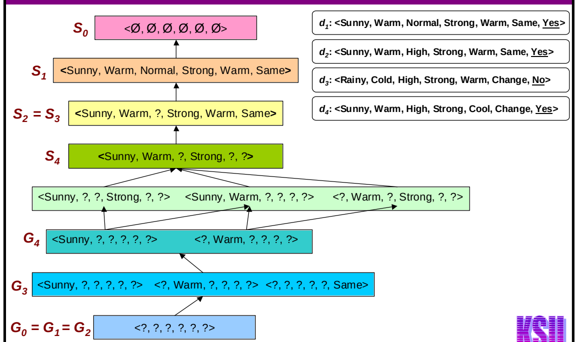
Add to *G* all minimal specializations *h* of *g* such that

1. *h* is consistent with *d*
2. Some member of *S* is more specific than *h*

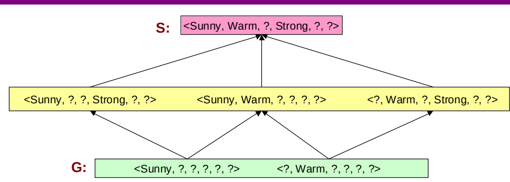
(These are the least upper bounds, or *joins*, $g \wedge d$, in $VS_{H,D}$)

Remove from *G* any hypothesis that is less general than another hypothesis in *G* (remove any dominating elements)

Example Trace



What Next Training Example?



- **What Query Should The Learner Make Next?**
- **How Should These Be Classified?**
  - **<Sunny, Warm, Normal, Strong, Cool, Change>**
  - **<Rainy, Cold, Normal, Light, Warm, Same>**
  - **<Sunny, Warm, Normal, Light, Warm, Same>**

What Justifies This Inductive Leap?

- **Example: Inductive Generalization**
  - **Positive example: <Sunny, Warm, Normal, Strong, Cool, Change, Yes>**
  - **Positive example: <Sunny, Warm, Normal, Light, Warm, Same, Yes>**
  - **Induced *S*: <Sunny, Warm, Normal, ?, ?, ?>**
- **Why Believe We Can Classify The Unseen?**
  - **e.g., <Sunny, Warm, Normal, Strong, Warm, Same>**
  - **When is there enough information (in a new case) to make a prediction?**

Terminology

- **Supervised Learning**
  - **Concept - function from observations to categories (so far, boolean-valued: +/-)**
  - **Target (function) - true function *f***
  - **Hypothesis - proposed function *h* believed to be similar to *f***
  - **Hypothesis space - space of all hypotheses that can be generated by the learning system**
  - **Example - tuples of the form *<x, f(x)>***
  - **Instance space (*aka* example space) - space of all possible examples**
  - **Classifier - discrete-valued function whose range is a set of class labels**
- **The Version Space Algorithm**
  - **Algorithms: *Find-S*, *List-Then-Eliminate*, candidate elimination**
  - **Consistent hypothesis - one that correctly predicts observed examples**
  - **Version space - space of all currently consistent (or *satisfiable*) hypotheses**
- **Inductive Learning**
  - **Inductive generalization - process of generating hypotheses that describe cases not yet observed**
  - **The inductive learning hypothesis**

Summary Points

- **Concept Learning as Search through *H***
  - **Hypothesis space *H* as a state space**
  - **Learning: finding the correct hypothesis**
- **General-to-Specific Ordering over *H***
  - **Partially-ordered set: Less-Specific-Than (More-General-Than) relation**
  - **Upper and lower bounds in *H***
- **Version Space Candidate Elimination Algorithm**
  - ***S* and *G* boundaries characterize learner's uncertainty**
  - **Version space can be used to make predictions over unseen cases**
- **Learner Can Generate Useful Queries**
- **Next Lecture: When and Why Are Inductive Leaps Possible?**