

For Monday

- Finish chapter 18
- Homework:
 - Chapter 18, exercises 1-2

Program 3

- Any questions?

Performance

- How can we measure performance?
- That is, what kinds of things do we want to **get** out of the learning process, and how do we tell whether we're getting them?

Performance Measures

- Classification accuracy
- Solution correctness and quality
- Speed of performance

Why Study Learning?

- (Other than your professor's interest in it)

Study Learning Because ...

- We want computer systems with new capabilities
 - Develop systems that are too difficult or impossible to construct manually because they require specific detailed knowledge or skills tuned to a particular complex task (**knowledge acquisition bottleneck**).
 - Develop systems that can automatically adapt and customize themselves to the needs of individual users through experience, e.g. a personalized news or mail filter, personalized tutoring.
 - Discover knowledge and patterns in databases, **data mining**, e.g. discovering purchasing patterns for marketing purposes.

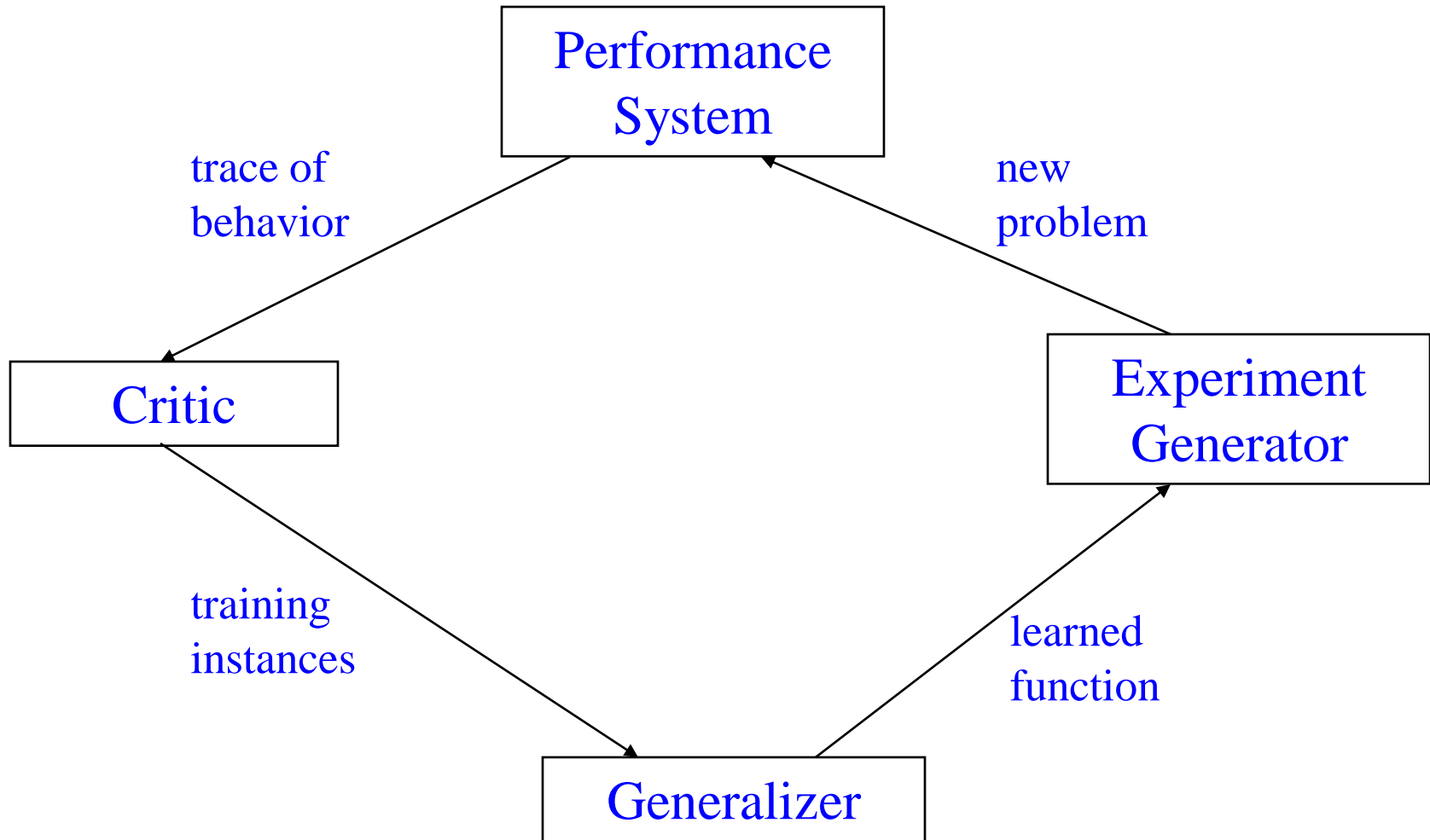
Study Learning Because ...

- Understand human and biological learning and teaching better.
 - Power law of practice.
 - Relative difficulty of learning disjunctive concepts.
- Time is right:
 - Initial algorithms and theory in place.
 - Growing amounts of on-line data.
 - Computational power available.

Designing a Learning System

- Choose the **training experience**.
- Choose what exactly is to be learned, i.e. the **target function**.
- Choose how to **represent** the target function.
- Choose a **learning algorithm** to learn the target function from the experience.
- Must distinguish between the **learner** and the **performance element**.

Architecture of a Learner



Training Experience Issues

- Direct or Indirect Experience
 - Direct: Chess boards labeled with correct move extracted from record of expert play.
 - Indirect: Potentially arbitrary sequences of moves and final games results.
- Credit/Blame assignment:
 - How do we assign blame to individual choices or moves when given only indirect feedback?

More on Training Experience

- Source of training data:
 - “Random” examples outside of learner’s control (negative examples available?)
 - Selected examples chosen by a benevolent teacher (near misses available?)
 - Ability to query oracle about correct classifications.
 - Ability to design and run experiments to collect one's own data.
- Distribution of training data:
 - Generally assume training data is representative of the examples to be judged on when tested for final performance.

Supervision of Learning

- Supervised
- Unsupervised
- Reinforcement

Concept Learning

- The most studied task in machine learning is inferring a function that classifies examples represented in some language as members or non-members of a concept from pre-classified training examples.
- This is called **concept learning**, or **classification**.

Simple Example

Example	Size	Color	Shape	Class
1	small	red	circle	positive
2	big	red	circle	positive
3	small	red	triangle	negative
4	big	blue	circle	negative

Concept Learning Definitions

- An **instance** is a description of a specific item. X is the space of all instances (**instance space**).
- The **target concept**, $c(x)$, is a binary function over instances.
- A **training example** is an instance labeled with its correct value for $c(x)$ (positive or negative). D is the set of all training examples.
- The **hypothesis space**, H , is the set of functions, $h(x)$, that the learner can consider as possible definitions of $c(x)$.
- The goal of concept learning is to find an h in H such that for all $\langle x, c(x) \rangle$ in D , $h(x) = c(x)$.

Sample Hypothesis Space

- Consider a hypothesis language defined by a conjunction of constraints.
- For instances described by n features consider a vector of n constraints, $\langle c_1, c_2, \dots, c_n \rangle$ where each c_i is either:
 - $?$, indicating that any value is possible for the i th feature
 - A specific value from the domain of the i th feature
 - \emptyset , indicating no value is acceptable
- Sample hypotheses in this language:
 - $\langle \text{big}, \text{red}, ? \rangle$
 - $\langle ?, ?, ? \rangle$ (most general hypothesis)
 - $\langle \emptyset, \emptyset, \emptyset \rangle$ (most specific hypothesis)

Inductive Learning Hypothesis

- Any hypothesis that is 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.
 - Assumes that the training and test examples are drawn from the same general distribution.
 - This is fundamentally an unprovable hypothesis unless additional assumptions are made about the target concept.

Concept Learning As Search

- Concept learning can be viewed as searching the space of hypotheses for one (or more) consistent with the training instances.
- Consider an instance space consisting of n binary features, which therefore has 2^n instances.
- For conjunctive hypotheses, there are 4 choices for each feature: T, F, \emptyset , ?, so there are 4^n syntactically distinct hypotheses, but any hypothesis with a \emptyset is the empty hypothesis, so there are $3^n + 1$ semantically distinct hypotheses.

Search cont.

- The target concept could in principle be any of the 2^{2^n} (2 to the 2 to the n) possible binary functions on n binary inputs.
- Frequently, the hypothesis space is very large or even infinite and intractable to search exhaustively.

Learning by Enumeration

- For any finite or countably infinite hypothesis space, one can simply enumerate and test hypotheses one by one until one is found that is consistent with the training data.

```
For each  $h$  in  $H$  do
  initialize consistent to true
  For each  $\langle x, c(x) \rangle$  in  $D$  do
    if  $h(x) \neq c(x)$  then
      set consistent to false
  If consistent then return  $h$ 
```

- This algorithm is guaranteed to terminate with a consistent hypothesis if there is one; however it is obviously intractable for most practical hypothesis spaces, which are at least exponentially large.

Finding a Maximally Specific Hypothesis (FIND-S)

- Can use the generality ordering to find a most specific hypothesis consistent with a set of positive training examples by starting with the most specific hypothesis in H and generalizing it just enough each time it fails to cover a positive example.

Initialize $h = \langle \emptyset, \emptyset, \dots, \emptyset \rangle$

For each positive training instance x

 For each attribute a_i

 If the constraint on a_i in h is satisfied by x

 Then do nothing

 Else If $a_i = \emptyset$

 Then set a_i in h to its value in x

 Else set a_i to "?"

Initialize *consistent* := true

For each negative training instance x

 if $h(x)=1$ then set *consistent* := false

If *consistent* then return h

Example Trace

$h = \langle \emptyset, \emptyset, \emptyset \rangle$

Encounter $\langle \text{small, red, circle} \rangle$ as positive

$h = \langle \text{small, red, circle} \rangle$

Encounter $\langle \text{big, red, circle} \rangle$ as positive

$h = \langle ?, \text{red, circle} \rangle$

Check to ensure consistency with any
negative examples:

Negative: $\langle \text{small, red, triangle} \rangle$ ✓

Negative: $\langle \text{big, blue, circle} \rangle$ ✓

Comments on FIND-S

- For conjunctive feature vectors, the most specific hypothesis that covers a set of positives is unique and found by FIND-S.
- If the most specific hypothesis consistent with the positives is inconsistent with a negative training example, then there is no conjunctive hypothesis consistent with the data since by definition it cannot be made any more specific and still cover all of the positives.

Example

Positives: <big, red, circle>,
 <small, blue, circle>

Negatives: <small, red, circle>

FIND-S -> <?, ?, circle> which matches
negative

Inductive Bias

- A hypothesis space that does not include every possible binary function on the instance space incorporates a bias in the type of concepts it can learn.
- Any means that a concept learning system uses to choose between two functions that are both consistent with the training data is called inductive bias.

Forms of Inductive Bias

- Language bias:
 - The language for representing concepts defines a hypothesis space that does not include all possible functions (e.g. conjunctive descriptions).
- Search bias:
 - The language is expressive enough to represent all possible functions (e.g. disjunctive normal form) but the search algorithm embodies a preference for certain consistent functions over others (e.g. syntactic simplicity).

Unbiased Learning

- For instances described by n attributes each with m values, there are m^n instances and therefore 2^{m^n} possible binary functions.
- For $m=2$, $n=10$, there are 3.4×10^{38} functions, of which only $59,049$ can be represented by conjunctions (a small percentage indeed!).
- However unbiased learning is futile since if we consider all possible functions then simply memorizing the data without any effective generalization is an option.

Lessons

- Function approximation can be viewed as a search through a pre-defined space of hypotheses (a representation language) for a hypothesis which best fits the training data.
- Different learning methods assume different hypothesis spaces or employ different search techniques.

Varying Learning Methods

- Can vary the representation:
 - Numerical function
 - Rules or logical functions
 - Nearest neighbor (case based)
- Can vary the search algorithm:
 - Gradient descent
 - Divide and conquer
 - Genetic algorithm

Evaluation of Learning Methods

- **Experimental**: Conduct well controlled experiments that compare various methods on benchmark problems, gather data on their performance (e.g. accuracy, run-time), and analyze the results for significant differences.
- **Theoretical**: Analyze algorithms mathematically and prove theorems about their computational complexity, ability to produce hypotheses that fit the training data, or number of examples needed to produce a hypothesis that accurately generalizes to unseen data (sample complexity).

Empirical Evaluation

- Training and Testing
- Leave-One-Out
- Cross-validation
- Learning Curves