

1 Final Review

1.1 Introduction

Textbook: Section 1.1

Final: 1 short question

What is Artificial Intelligence?

	Human	Rational (Ideal)
Thought	Thinking like a human	Thinking rationally
Behavior	Acting like a human	Acting rationally

Thinking like a human:

Solving problems using the same thought processes as another intelligent agent, namely humans. Closely tied to cognitive science, or the study of how humans think.

Thinking rationally:

Rather than rely on the (flawed) intelligence of humans, this view of AI prefers to use formally proven models of thought like logic as the model of intelligence.

Acting like a human:

This view is captured in the Turing test and Total Turing test.

Acting rationally:

Acting correctly relative to a predefined set of goals.

1.2 Search

Textbook: Chapter 3

Final: 1 in-depth question

Textbook: Sections 4.1-4.3

Final: 1 short question

A search problem is made up of (3 things):

1. initial state
2. successor function
3. goal description (test)

Search strategies differ in the way they...

expand the fringe nodes

Name the search strategies (9):

1. A^*
2. BFS

3. Best-first search
4. Bi-directional
5. DFS
6. Depth limited
7. Greedy
8. Iterative deepening
9. *IDA**

What makes a heuristic admissible?
never overestimates

What is the difference between iterative improvement and search algorithms?
Iterative improvement algorithms do not keep track of the path from the initial state to the current node.

Name the iterative improvement algorithms (4):

1. Hill-climbing
2. Simulated annealing
3. Local beam search
4. Genetic Algorithms
5. *Ant Colony Optimization (New)*

1.3 Games

Textbook: Sections 6.1 and 6.2

Final: 1 short question

What is the difference between games and other search problems?
Your opponent(s)

A game is defined by (4 things):

1. initial state
2. successor function
3. end-game description (test)
4. utility function (determines quality of end-game states for each player)

Name the two game algorithms:

1. MINIMAX

2. Alpha-Beta

Why is Alpha-Beta better?

Prunes the search tree

What makes a game nondeterministic?

An element of chance (any game with dice, card games)

When are evaluation functions used?

When you don't have time to search the entire tree.

1.4 Logic

Textbook: Sections 8.1 and 8.2

Final: 1 short question

A predicate is either *TRUE* or *FALSE*.

Syntax is made up of: *symbols, logical operators, and grammar*

Semantics give the: *meaning*

Give the inference rule Modes Ponens

$$\frac{a \rightarrow b}{a}$$

What is this equivalent to?

$$\neg(a \vee b) \Leftrightarrow (\neg a \wedge \neg b)$$

And this?

$$(a \wedge (b \vee c)) \Leftrightarrow ((a \wedge b) \vee (a \wedge c))$$

Order of operations:

$$\neg, \wedge, \vee, \rightarrow, \leftrightarrow$$

So $\neg p \vee q \wedge r \rightarrow s$ is equivalent to $((\neg p) \vee (q \wedge r)) \rightarrow s$

What two kinds of objects exist in first-order logic?

Constants and variables

A relation is either *TRUE* or *FALSE*.

A function returns an *object*.

1.5 Learning

Textbook: Sections 18.1 and 18.2

Final: 1 short question

Textbook: Section 18.3

Final: 1 in-depth and 1 short question

Textbook: Section 20.5

Final: 1 in-depth question

Learning is a process that searches for the most likely hypothesis given a set of observations (data or knowledge).

Typical goals: learn a new concept or behavior, or improve the performance (speed or solution quality) of an existing technique

Def: Hypothesis. *A possible explanation for a set of observations.*

Def: Training data. *The observations used during the learning process.*

Def: Test data. *Observations used to evaluate the learned solution (cannot overlap with training data).*

Overfitting:

Producing a solution that has very little error for the training set, but a lot of error when applied to new (test) examples.

Ockham's Razor:

The simplest hypothesis that fits a set of observations is usually the correct one

The choice of hypothesis space and preferences among its elements is called a *learning bias*.

Important factors in a learning problem:

1. What is learned?
2. What representation is used to record the observations?
3. How much and what kind of feedback is available?

Feedback:

Supervised - *any measure of performance*

Unsupervised - *no feedback, used to find patterns*

Reinforcement - *indirect measures of performance*

Build a decision tree for the following data:

Class	Rarity	Age	Wear
TRUE	rare	new	low
TRUE	rare	old	low
TRUE	common	old	low
FALSE	rare	old	high
FALSE	common	new	low
FALSE	common	new	high

$$\text{cost}(p, n) = -\frac{p}{p+n} \cdot \lg \frac{p}{p+n} - \frac{n}{p+n} \cdot \lg \frac{n}{p+n}$$

$$\text{Gain} = \text{cost}(p, n) - \left(\frac{p_1 + n_1}{p+n} \cdot \text{cost}(p_1, n_1) + \dots + \frac{p_k + n_k}{p+n} \cdot \text{cost}(p_k, n_k) \right)$$

Answer: Information gain of each attribute at root:

$$\text{Gain}(\text{rarity}) = \text{cost}(3/6, 3/6) - (3/6) * \text{cost}(2/3, 1/3) - (3/6) * \text{cost}(1/3, 2/3) = 0.082$$

$$\text{Gain}(\text{age}) = \text{cost}(3/6, 3/6) - (3/6) * \text{cost}(1/3, 2/3) - (3/6) * \text{cost}(2/3, 1/3) = 0.082$$

$$\text{Gain}(\text{wear}) = \text{cost}(3/6, 3/6) - (4/6) * \text{cost}(3/4, 1/4) - (2/6) * \text{cost}(0,1) = 0.459$$

Use *wear* as the root's split point. High wear is always FALSE, so this leaves the following data:

Class	Rarity	Age	Wear
TRUE	rare	new	low
TRUE	rare	old	low
TRUE	common	old	low
FALSE	common	new	low

$$\text{Gain}(\text{rarity}) = \text{cost}(3/4, 1/4) - (2/4) * \text{cost}(1, 0) - (2/4) * \text{cost}(1/2, 1/2) = 0.311$$

$$\text{Gain}(\text{age}) = \text{cost}(3/4, 1/4) - (2/4) * \text{cost}(1/2, 1/2) - (2/4) * \text{cost}(1, 0) = 0.311$$

Either split will work.

Problems one can encounter when working with Decision Trees:

1. Insufficient training data. Valid values for some attributes do not appear.
2. Insufficient attributes.
3. Overfitting. Data can be noisy, or the above case is true.
4. Limited expressiveness. Real-valued attributes are difficult to handle as are problems like parity.

Neural networks are based on an analogy with *the human brain*.

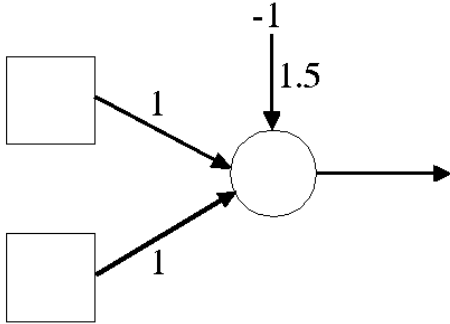
What is a perceptron?

A neural network where input nodes are connected directly to output nodes, i.e. there are no hidden nodes.

What can a perceptron learn?

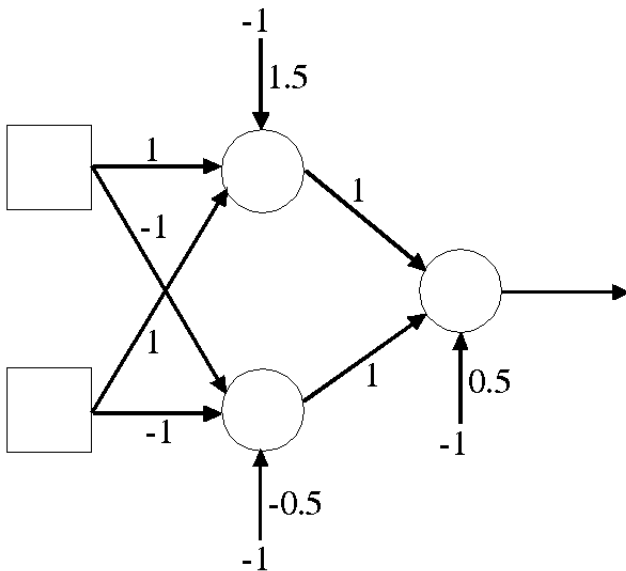
A perceptron can learn linear separations of n -dimensional spaces, where n is the number of inputs.

What does this network do?



Answer: AND

How about this one?



Answer: Even Parity

Name some similarities between Decision Trees and Neural Networks.

Both learn from examples (inductive learning), use supervised learning, and learn a classification scheme.

Name some differences between Decision Trees and Neural Networks.

Neural networks learn real-values functions from numeric input whereas decision trees learn from categorized data. Neural networks reuse training examples over multiple epochs, where decision trees can only use a training example once.

1.6 Knowledge Representation

Textbook: Section 10.1 and 10.2

Final: 1 short question

Ontology: *an explicit formal specification of how to represent the objects, concepts, and other entities that are assumed to exist in some area of interest and the relationships that hold among them*

What is inheritance?

A link in an ontology indicating that a smaller group (category) of objects are a special case of a larger group (category) of objects.

How does it help in building ontologies?

Since objects in the smaller group inherit attributes from their parents, this condenses the information in an ontology. Smaller ontologies can capture large amounts of information.

Form an ontology using the following musical styles:

European, classical, romantic, rock, modern, Indian, jazz, oriental

(You can do this one on your own.)

1.7 Expert systems

Expert Systems Notes

Final: 1 short question

CLIPS Tutorial

Final: 1 in-depth question

What are the typical uses of expert systems?

1. Provide advice for non-expert humans (e.g. online self-help)
2. Support the decision-making process of human-experts (decision-support systems)
3. Enable intelligent agents to perform tasks that require expert knowledge and/or common sense (e.g. natural language processing)

Three main components of expert systems:

1. Knowledge-base
2. Inference engine
3. I/O Module

What algorithms does the inference engine use?

forward-chaining and/or backward-chaining

What goes into an expert system shell?

Inference engine, I/O module

What do these CLIPS commands represent?

```
CLIPS> (assert (lawyer Claudia))
CLIPS> (assert (works-for-advocacy-firm Claudia))
CLIPS> (assert (lawyer Frank))
CLIPS> (assert (friends Claudia Frank))
```

```
CLIPS> (defrule rich-lawyers
CLIPS>      (lawyer ?x)
CLIPS> =>
CLIPS>      (assert (rich ?x))
CLIPS> )
```

```
CLIPS> (defrule advocacy-lawyers
CLIPS>      ?rule1 <- (rich ?x)
CLIPS>      (works-for-advocacy-firm ?x)
CLIPS> =>
CLIPS>      (retract ?rule1)
CLIPS> )
```

```
CLIPS> (defrule richPeoplesHouses
CLIPS>      (rich ?x)
CLIPS> =>
CLIPS>      (assert (hasBigHouse ?x))
CLIPS> )
```

```
CLIPS> (defrule advocacyHouses
CLIPS>      (works-for-advocacy-firm ?x)
CLIPS> =>
CLIPS>      (assert (hasSmallHouse ?x))
CLIPS> )
```

```
CLIPS> (defrule visiting
CLIPS>      (hasSmallHouse ?x)
CLIPS>      (not (rich ?x))
CLIPS>      (friends ?x ?y)
CLIPS>      (hasBigHouse ?y)
CLIPS> =>
CLIPS>      (assert (oftenVisits ?x ?y))
CLIPS> )
```

Answer – In FOL:

- Lawyer(Claudia)
- Works-for-advocacy-firm(Claudia)
- Lawyer(Frank)
- Friends(Claudia, Frank)

- $\forall x \text{ Lawyer}(x) \rightarrow \text{IsRich}(x)$
- $\forall x \text{ Works-for-advocacy-firm}(x) \rightarrow \neg \text{IsRich}(x)$
- $\forall x \text{ IsRich}(x) \rightarrow \text{HasBigHouse}(x)$
- $\forall x \text{ Works-for-advocacy-firm}(x) \rightarrow \text{HasSmallHouse}(x)$
- $\forall x, y \text{ HasSmallHouse}(x) \wedge \neg \text{IsRich}(x) \wedge \text{Friends}(x, y) \wedge \text{HasBigHouse}(y) \rightarrow \text{OftenVisits}(x, y)$

1.8 Planning

Textbook: Section 11.1–11.3

Final: 1 short question

Goal: find a sequence of tasks that leads to a desired situation (goal state)

What do expert systems and planning algorithms have in common?

Both use subsets of FOL. As a result both can use generic algorithms to solve a broad range of problems encapsulated in formal logical statements.

What is the basic language used to represent problems in planning?

STRIPS

In STRIPS a world is represented as a collection of *literals*.

Examples: *door(1,2), in(A,1)*

An operator is defined by *preconditions* and *effects*.

Example:

go(x,y)

where *x,y* are rooms

Pre: *robot-in(x) \wedge door(x,y) \wedge \neg locked(x,y)*

Eff: *del robot-in(x)*

add robot-in(y)

Three main search strategies for planning:

1. Forward search
2. Backward search
3. Partial order planning

What is the advantage of partial order planning over the other two strategies?

Much lower branching factor

Which of these does Prodigy do?

All three

What is hierarchical planning?

When high-level plans are refined by creating a new sub-plan for each operator using more specific operators.

1.9 Emotions in AI and Robotics

Cindy Bethel's ppt presentation

Final: 1 short question

Go over the review questions included in Cindy Bethel's presentation.

1.10 Uncertainty

Class notes

Final: 1 short question

Given an abstract solution (ignores uncertainty) that responds to changes, how are these changes typically handled?

1. Redo from scratch (no changes to algorithm)
2. Generate robust solutions (change goal of algorithm)
3. Focus on efficient repair (add the ability to repair solutions)

What are the three main theories used to handle uncertainty when an abstract solution won't work?

1. Probability theory
2. Dempster-Shafer theory
3. Fuzzy Logic

What is Bayes' rule used for in AI?

Updating existing probability models as new information is gathered.

What is a Bayesian network?

A formal description of a problem that describes the probabilistic relationships between relevant statements (random variables).

Which of the above theories handles uncertainty?

Probability theory and Dempster-Shafer theory

Which of the above theories handles ignorance?

Dempster-Shafer theory

Which handles impreciseness?

Fuzzy Logic