David Keil, Framingham State University

CSCI 300 Artificial Intelligence

5. Supervised learning and natural language

1. Supervised learning
2. Symbol-based learning
3. Connectionist learning
4. Evolutionary computation
5. Natural-language processing

Inquiry

- How do you learn
  - facts – critical thinking
  - skills – problem solving
- How might (a) NAO or (b) Siri operate as a *learning* agent?
- What type of learning occurs in *static environments*?
- Do we learn by generalizing from what we see?

Topic objective

5. Describe ways to supervise agents to learn and improve their behavior

Subtopic outcomes

5.1 Explain what *learning* is*

5.2a Describe methods of symbol-based *supervised learning*

5.2b Apply the decision tree learning method to sample data

5.3a Describe the *connectionist* approach to AI*

5.3b Construct and train a perceptron

5.4 Describe evolutionary computation

5.5 Explain concepts of *natural-language processing*

1. Supervised learning

- What has learning been for you?
- Do you learn from *patterns* in what *you* observe?
- How did you learn
  - to drive?
  - what music genre you like?
  - what kind of job you want to apply for?

What is learning?

- *Definition* (H. Simon, 1983): "Any change in a system that allows it to perform better the second time on repetition of the same task or on another task drawn from the same population."
- Hence learner must *generalize* from limited experience
- All learning may be seen as a state-space search for a set of appropriate actions to use in response to percepts

Learning as performance improvement

- An agent *learns* from experience *E*, w.r.t. task set *T*, using performance measure *P*, iff agent's performance measure under *P*, on tasks in *T*, improves with experience *E*
- *Example:* Driving
  - *T*: driving using vision sensors
  - *P*: average distance without collision
  - *E*: images (percepts) paired with steering commands (actions)

Kinds of learning

- We classify learning as *supervised* (topic 5) or *interactive* (topics 6-7)
- Three types:
  - *Symbol-based*; e.g., generalizations
  - *Connectionist*: neural networks, inspired by brain's changing of thresholds for firing
  - *Emergent*: evolutionary-inspired techniques (topic 7)

Supervised learning may yield policies

- A decision-theoretic agent may be given a policy using training data
- (See section 4.5 for decision-theory discussion)

Supervised learning

- *Principle:* use training data to improve future ability to act; e.g., policy
- *Useful to learn:*
  - state-action mapping
  - inference from percept to environment state
  - model of how environment changes
  - results of possible actions
  - utilities of states
  - utilities of actions

Rote memorizing vs. learning

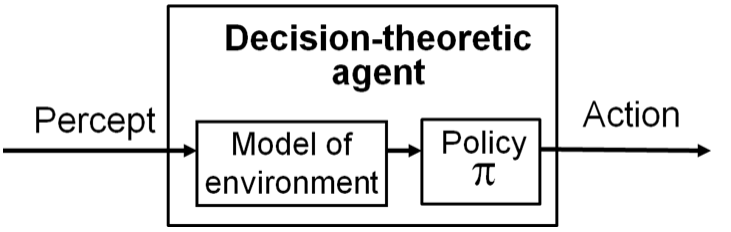
- Some education focuses on memorizing
- It involves reading or listening and choosing some words to remember
- It is more processing-intensive than loading data from a device or port
- But it isn't what AI research calls *learning*
- Constructivist theory of learning: (human) learning is *the learner's construction of knowledge*

Concept learning

- *Domain:* a set X of possible instances of concept
- Target concept: $c : X \rightarrow \{ \text{T}, \text{F} \}$
- *Training examples:* a set of pairs (x, y), with $x \in X$, $y = c(x)$
- Where $y = \text{F}$, example is "negative"
- Where $y = \text{T}$, example is "positive"
- *Goal of learning:* to find c by generalizing from training examples
- *Method:* Search for a hypothesis h s.t. $h(x) = c(x)$ for all $x \in X$

Decision-theoretic agents

- Policy is derived from a model of the environment
- Best actions are those that bring agent to high-utility states for maximum long-term reward



Value iteration algorithm

- Calculates utilities of states, allowing choice of optimal action for each state
- Utility of state s under policy π $U_\pi(s)$ depends on reward $R(s)$, but utility and reward are distinct, in that U is long-term expected reward
- *Utility of a state:* $R(s)$, plus expected discounted utility of the next state under optimal choice, using discount value γ
- Evolves a table U of estimated utilities of states

Value-iteration(E, γ)

$U' \leftarrow (0,0,0,0,0,0,0,0,0)$
repeat
  $U \leftarrow U'$
  $\delta \leftarrow 0$
  For each $s \in S$
    $U'[s] \leftarrow E.R(s) + \gamma \max_{a \in A} E.P_{s'}(s' \mid s, a)\, U[s']$
    if $|U'[s] - U[s]| > \delta$
      $\delta \leftarrow |U'[s] - U[s]|$
until $\delta < \varepsilon(1 - \gamma) / \gamma$
return U

U is utility estimate
A is action set
$R(s)$ is reward
$P_{s'}(s \mid s, a)$ is probability of transition from s' to s on action a
γ Is discount value

R

0	0	+10
0	0	0
0	0	0

- After value iteration, *policy* may be set by choosing actions that favor high-valued (high-utility) states

Subtopic outcome

5.1 Explain what learning is*

2. Symbol-based learning

- Do you learn from *patterns* in what *you* observe?
- How did you learn
  - to drive?
  - what music genre you like?
  - what kind of job you want to apply for?

Inductive inference

- A deterministic supervised-learning algorithm
- Given examples of concept *c*, return an approximation *h* of *c*
- *h* is a hypothesis; it may be subject to improvement
- Inductive inference applies *Ockham's Razor*: prefer the simplest hypothesis that fits the data

Logic-based abduction

- An *abductive explanation* of a set of observations O may be defined as
  - a minimal set of hypotheses H
  - consistent with background knowledge set K
  - where $H \cup K$ entails O
- *Abduce*(K, O) = H iff
  $\neg(O \rightarrow K) \wedge (O \rightarrow H) \wedge$ consistent($H \cup K$) $\wedge$
  (no subset of H has the previous three properties)
- *Abduction* back-chains from conclusions (observations) to antecedents (hypotheses)

Abductive inference: $((p \rightarrow q) \wedge q) \rightarrow p$

- Strictly *unsound* because not always true; but quite often applicable in practice
- *Example:* excess sugar produces cavities, and I have a cavity, so I must be eating excess sugar
- It is valid to reason this way with a certain degree of confidence
- $p \rightarrow q$ (0.9) means if p is believed to be true then it is believed that q will hold 90% of the time
- Abduction helps us to see patterns of causality in the world

Set cover based abduction

- Abduction is "reasoning to the best explanation" for the presence of an item of data
- If $(p \rightarrow q) \wedge q$, then a plausible *explanation* for q is p
- *Set cover approach:* an exhaustive covering of predicates describing observations by use of predicates denoting hypotheses
- *Basis:* let $R \subseteq Hypotheses \times Observations$ where R is causal

Set cover example

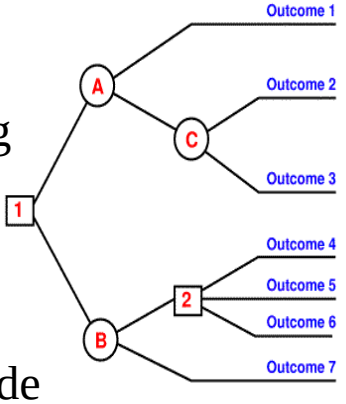
- Let S be all possible observations; let T be …
- Let set $S_2 \subseteq S$ be a set of observations,
- Let H be a set of hypotheses sufficient to explain S_2
- *Set cover:* Given $R \subseteq S \times T$, $S' \subseteq S$ is a set cover w.r.t. R if every t in T is in a pair that is in R, i.e.:

  $(\forall t \in T)\ (\exists s \in S')\ (s, t) \in R$
- [Need a more concrete example; want to identify T]

Information theory and AI (exploration)

- In exploring an environment or data set, we want to act first toward finding a solution quickly
- Some answers to questions give us more information than others
- *Example:* In assessing credit risk, three attributes correlate highly to risk: debt, bad history, inverse of income, inverse of collateral
- Information in a communication can be quanti-fied as the *amount of surprise* (new information)

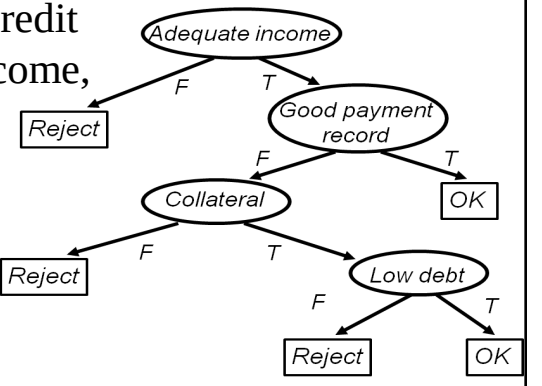
Decision trees



- May diagram any branching algorithmic process
- Root is starting point; leaves are terminations
- Each branching step is a node
- Edges reflect alternative choices
- Depth of tree expresses algorithm's running time

Decision trees as policies

- A *decision tree* may perform a series of binary tests resulting in a yes/no answer
- *Input:* a set of attributes; *Output:* a decision (box)
- *Example:* Granting credit requires adequate income, plus a good payment record or both collateral and low debt



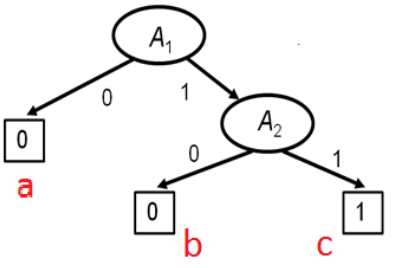
Decision tree learning

- Examples are given in a *training set*
- DTL classifies instances of concepts in trees whose leaves are classifications, e.g., yes/no
- Discrete outputs occur in *classification learning*, continuous outputs in *regression learning*
- DTL is robust to errors, unlike other concept-learning algorithms

Decision-tree learning

- Given *example data* showing results of decision-tree execution for combinations of attribute values, decision trees may be *induced* (learned)
- *Example:* The table below gives result values for 3 attributes *A* and 5 causes *x*; corresponding decision tree is at right

	Attributes			
Examples	A_1	A_2	A_3	***y***
x_1	0	0	1	0
x_2	1	0	1	0
x_3	0	1	0	0
x_4	1	1	1	1
x_5	1	1	0	1



Explanation of the above DT

- Result value *a* is 0 because both cases (x_1 and x_3) where A_1 was 0 have result 0
- Result value *b* is 0 because both cases (x_1 and x_2) where A_2 was 0 have result 0
- Result value *c* is 1 because both cases (x_4 and x_5) where A_2 was 1 have result 1

Computational learning theory

- *Probably approximately correct* (PAC) learning: any hypothesis consistent with a sufficiently large training set is unlikely to be very mistaken
- For hypothesis h, error (h) = $P(h(x) \neq f(x) \mid x$ in training set distribution)
- Approximately correct h is h s.t. error(h) $\leq \varepsilon$, a small constant

Learning logical assertions from examples

- A form of inductive learning
- The universe of hypotheses is a set of logical assertions
- Hypotheses inconsistent with the example set are ruled out – false negatives or positives
- This elimination of hypotheses is analogous to the resolution rule
- For each example, generalize hypothesis if false negative is found, specialize it if false positive

Version space learning

- *Least commitment* approach seeks refinement of set of all hypotheses consistent with a growing set of examples
- This version space is the set bounded by the most specific and most general sets consistent with all examples so far
- *Drawbacks:* version space will collapse in case of noise
- Not practical for most real-world problems

Knowledge-based induction

- *Definition:* the cumulative development of knowledge used in inference
- *Hypothesis* $\wedge$ *Descriptions* |= *Classifications* (*entailment constraint*)
- *Example:* Traveler hearing Portuguese spoken infers that all inhabitants speak Portuguese, but meeting Fernando does not infer that all inhabitants are named Fernando

Subtopic outcomes

5.2a Describe and use methods of symbol-based supervised learning

5.2b Apply the decision tree learning method to sample data

3. Connectionist learning

- What is the brain like?
- Do your neurons reason?
- Are neurons smart?
- Can the mind be smart?

Connectionism

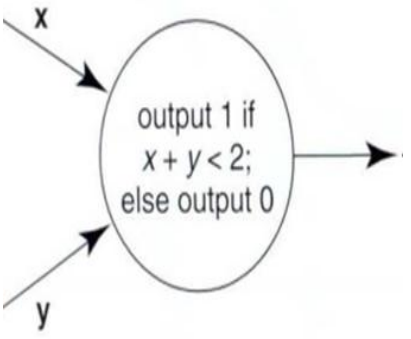
- Based on brain, a special-purpose organ oriented to navigation and recognition
- *Connectionist* systems are also known as *parallel distributed processing*
- Non symbolic; based on model of *neuron*
- *Applications*: perception related; classification; pattern recognition; memory recall; prediction; optimization; noise filtering

Artificial neural nets

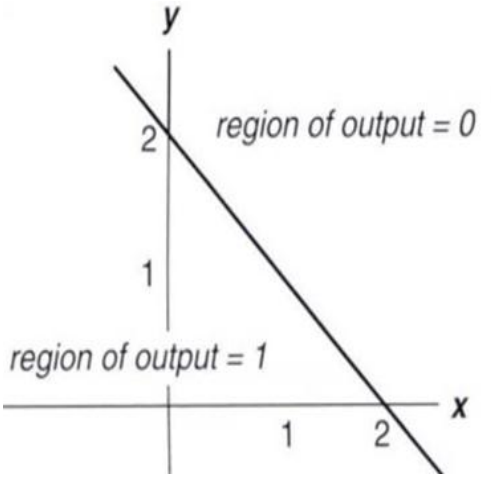
- *Artificial neural nets* are composed of simulated *neurons* with input and output
- Neuron fires (outputs 1) if sum of inputs passes a threshold
- *AND*, *OR*, *NOT* may be implemented with threshold activation functions

Example: fires if $x + y < 2$

Neuron

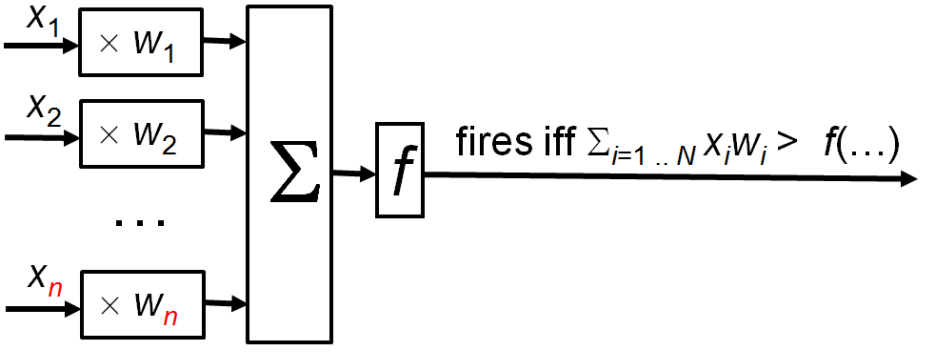


Function computed



Artificial neuron

- Inputs are $x_{1\,..\,n}$
- Weights are $w_{1\,..\,n}$
- f is threshold function
- Neuron fires when sum of weighted inputs exceeds threshold



Activation functions

- *Output activation* of a neuron i: $a_i = g(\Sigma_{j=0 \text{ to } n} W_{j,i} a_j)$, where a_j is output activation of neuron j, and $W_{j,i}$ is weight of synapse (j, i)
- *Activation function* may be threshold ($g(x) = 0$ if $x < k$, else 1) or *sigmoid*

Types of neural nets

- *Perceptrons* are single-layer
- Can only compute *linearly separable* functions; XOR for example is not LS (Minsky-Papert, 1969)
- *Feed-forward* (acyclic) nets are arranged in layers that pass signals to each other
- *Recurrent* (cyclic) has internal state
- *Multi-layer* nets have hidden neuron units

Perceptron example

- *Linear functions* may be computed by perceptrons; see topic 5 questions and sites referenced there

Perceptron learning

- Devised by Rosenblatt, 1958
- Single-layer neural net
- Inputs and outputs are in {–1, 1}
- Neurons begin learning process with default synapse weights
- Perceptrons perform an optimization search in *weight space*
- Perceptrons have a probabilistic interpretation in weight-update vector

Updating weights

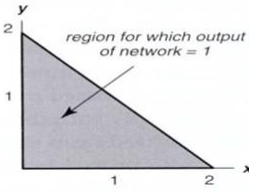
- In systems of connected components, such as the brain, learning consists of the adjusting of connections
- Sum of squares of errors is a standard measure used to update weights
- *Update:* $w_j \leftarrow w_i + \alpha \times \text{Err} \times g'(\text{in}) \times x_j$ where α is learning rate

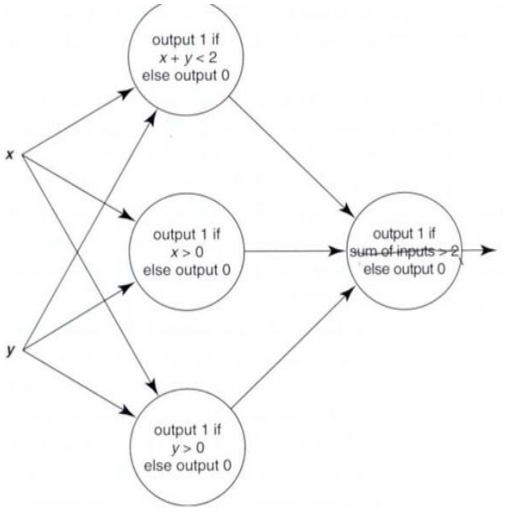
Weight adjustment

- To train a NN, sample inputs are provided, and weights are adjusted according to results of training tests
- $\Delta w_i = c(d - \text{sign}(\Sigma w_i - x_i))\, x_i$ where c is a constant, d is a desired output, *sign* is actual output
- Result of adjustment is a set of weights that minimizes average error over training set
- If some set of weights will generate correct output for all of training set, perceptron will find it

Multi-neuron network

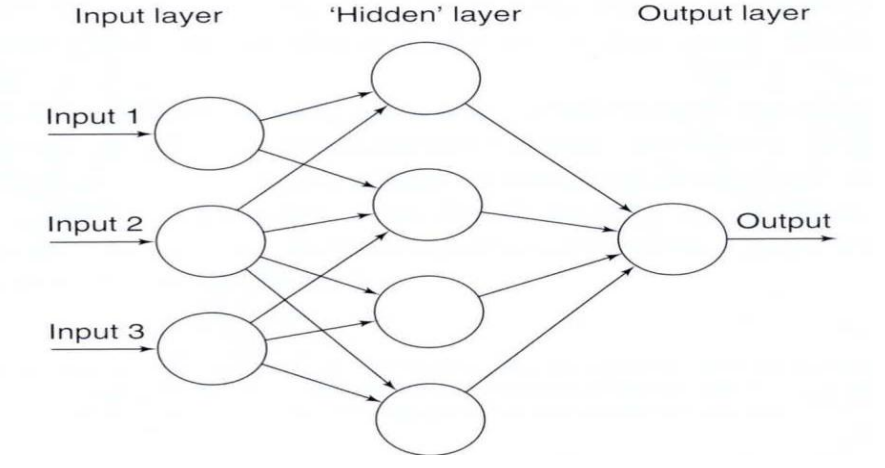
- A multi-neuron network may fire under a more complex condition
- *Example*: right, fires if $0 \leq (x + y) \leq 2$ (function shown below)





Hidden layers

- *Hidden layers* of neural nets are neurons that are neither external inputs nor external outputs



Backpropagation learning

- Overcomes limitation of perceptrons to linearly separable functions
- Adjusts weights

Handwriting-recognition learning

- The problem of recognizing digits may be stated as NIST standard 60,000 images paired with decimal digits
- *Neural nets* in different versions solve the problem, with error rates as low as 0.7%
- Optimal linear separator *Kernel machine* (support vector architecture) finds largest *margin* between separator and positive and negative examples
- *Shape matching* maps points in a pair of images that correspond to each other

Subtopic outcomes

5.3a Describe the *connectionist* approach to AI*

5.3b Construct and train a perceptron

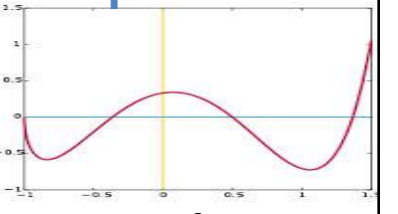
4. Evolutionary computation

- Do you ever solve a problem by guessing the answer and testing and refining your guesses?
- How does natural evolution work?

Evolutionary computation

- A probabilistic, population-based way to develop approximate solutions to difficult problems using notions of utility and fitness
- Modeled on natural evolution of species and behaviors of some life forms
- A growing research area encompassing genetic algorithms, evolutionary programming, evolution strategies, ant computing, swarm computing, particle swarm optimization
- EC is ontogenetic learning

The function-optimization problem



- Let $f : \mathbf{N}^k \rightarrow \mathbf{R}$ for some k
- *Problem*: Find some $x \in \mathbf{N}^k$ such that $f(x)$ is maximal
- *Example*: If x is the set of proportions of ingredients in a fuel mixture, then $f(x)$ is fuel cost-efficiency under this mixture
- *Optimizing* $f(x)$ means finding the most cost-efficient mixture
- For an *algorithm* to optimize a function, we must have $f : X \rightarrow Y$ with X, Y finite

Fitness and function optimization

- *Example*: Suppose *f* is viewed as a *fitness* function and *x* is the set of attributes of individuals of a population
- Then finding *x*, s.t. *f* (*x*) is maximal, is finding the *fittest* possible individual of the species, i.e., that with the best attributes to assure survival
- Evolution by natural selection tends to optimize fitness, over many generations
- Optimization problems in general are hard

Evolutionary computation

- A state space is explored generating and comparing a *population* of states rather than one state at a time
- *Evolutionary algorithm* repeatedly modifies and selects from a population of solutions
- Selection is driven by fitness test of each member of the population
- Randomization is used to explore the state space

The evolutionary algorithm

$t \leftarrow 0$
Initialize (P_0)
$V \leftarrow$ *Evaluate* (P_0)
While not *Terminate* (V, t) do
  $t \leftarrow t + 1$
  $P_t \leftarrow$ *Select* (P_{t-1}, V)
  $P_t \leftarrow$ *Alter* (P_t)
  $V \leftarrow$ *Evaluate* (P_t)
Return P_t

time

vector of fitness values

population at time *t*

Mutate or cross over

- *Evaluate* applies fitness function to individuals
- *Select* chooses some members of *P* to survive
- *Alter* changes *P* by mutation or crossover
- *Terminate* ends algorithm after a given goal or a time deadline is reached

Genetic algorithms

- At each step, the population is selected and regenerated using genetic operators, e.g., mutation and crossover
- *Parameters:* % of population to retain at each generation, which mate and produce offspring, which probability measures, if any, to use

Example: Checkers (Samuel, 1950s)

- Let fitness function $f : \mathbf{N}^k \rightarrow \mathbf{R}$ be an evaluator of board positions from a black or red viewpoint
- Let $x \in \mathbf{N}^k$ be a k-tuple of *weights* for each of k different criteria for evaluating a position
- *Example*: let x_1, x_2 be importance of number of kings, number of checkers threatened, etc.
- Then writing a good checkers-playing program reduces to finding a good set of relative weights $x_{1, \ldots,} x_k$ for these criteria
- *Result:* machine play at very high level

Subtopic outcome

5.4 Describe evolutionary computation

5. Natural-language processing

- What is language?
- How did you learn to understand speech?
- How did you learn to speak?

Human language

- *Speech act:* utterance for purpose of communication
- "Speech" is generic, as written, signed, etc.
- *Uniqueness of human language:* only it can specify an infinity of distinct messages
- Used in coordination of behavior: querying, informing, requesting actions, acknowledging, committing to act, declaring (as in, "Strike 3")

Steps of communication

- *Intention:* meaning sought to communicate
- *Generation:* plan of creation of utterance with meaning intended
- *Synthesis:* production of utterance from symbols
- *Perception:* hearing or recognition
- *Analysis:* syntactic, semantic, pragmatic breakdown, possibly yielding multiple interpretations
- *Disambiguation:* considering probabilities
- *Incorporation* into hearer's language

Languages

- *Language:* a set of strings, concatenating terminal symbols (e.g., words)
- *Grammar:* finite set of rules to specify a language
- *Semantics:* meaning
- *Pragmatics:* meaning in context

Formal language theory

- *Language:* a set of strings over a finite alphabet Σ of symbols, e.g., letters
- *Empty language* (with no strings): ∅
- *Null string* (with no symbols): λ
- *Operations on languages:* concatenation, union, intersection, complement
- Inductive definition of universal language Σ* of all strings over Σ: $\Sigma^* = \{xa \mid x \in \Sigma^*, a \in \Sigma\}$

Regular languages and regular expressions

- A RL is specified by a *regular expression* using any combination of these operations:
  - Concatenation
  - Selection (|, +, ∪)
  - Iteration (*) (binds to preceding symbol)
- *Examples:*
  - $(01)^*$: all strings that repeat "01", ≥ 0 times
  - $01^* \mid 10^*$: all strings that either consist of a 0 followed by zero or more 1's, or consist of a 1 followed by zero or more 0's

Context-free grammars

- *Definition:* CFG *G* consists of *terminal symbols; nonterminals* (NT, names); *production rules*
- *Nonterminal symbols:* symbols denoting phrase structures, defined by rules, e.g., $S \rightarrow NP \quad VP$
- *Production rules* are of the form $X \rightarrow Y\ Z\ldots$ where $X \in NT$ and $Y, Z\ldots \in (\Sigma \cup NT)$
- *Example:* $S \rightarrow \lambda \mid 0S1$
- By this grammar, a sentence can be a null string or a 0, followed by a sentence, followed by a 1

English-language grammar

- *Lexicon:* a set of words, with their parts of speech (N, V, adj, adv, article...)
- *Grammar:* a set of production rules:
  S → NP VP | S conj S
  NP → noun | article noun
  VP → verb | VP NP VP adj
- *Parsing* applies rules to a sentence, starting with sentence symbol (S), replacing left nonterminal with a definition string:
  S ⇒ NP VP ⇒ noun VP ⇒ noun verb ⇒
  John verb ⇒ John goes

Semantic interpretation

- May be the process of extracting a FOL sentence from a natural-language sentence
- *Examples:*
  - Sem("the wumpus") = Wumpus1
  - Sem("the wumpus is dead") = Dead (Wumpus1)
- *Lambda calculus:* "John loves Mary" = (λx *Loves*(x, *Mary*)) (*John*), where λx defines a predicate whose argument is John
- *Quantification:* "I sleep" = ($\exists e$) $e \in$ Sleep(Speaker) $\wedge$ During(*Now*, e)

Querying conceptual graphs

- *Sentence:* Fido likes June
- *Semantic representation:*
  [pic]
- *Query:* Who likes June?
- [pic]

Semantic ambiguity

- "Every dog has a day" =
  - $(\forall d)\ d \in Dogs \Rightarrow (\exists a)\ a \in Days \wedge (\exists e)\ e \in Has(d, a, Now)$
    or
  - $(\exists a)\ a \in Days \wedge (\forall d)\ d \in Dogs \Rightarrow Has(d, a, \text{Now})$
- "Each dog has its own day" vs. "There is a day shared by all dogs"
- *Ambiguity* may be lexical, syntactic, or semantic

Knowledge models

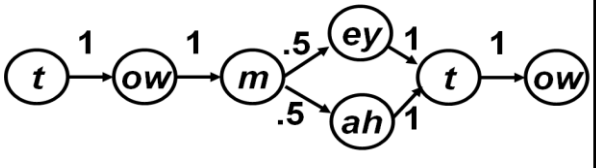
- *World:* proposition occurs in the world
- *Mental:* speaker intends to communicate the information to hearer
- *Language:* correspondence of word choice to fact
- *Acoustic:* generation of sounds corresponding to words

Discourse understanding

- *Discourse:* a string, usually more than one sentence
- *Issues:*
  - *Reference resolution:* interpretation of pronoun or other reference
  - *Coherence:* depends on order; issues: enabling, causation, explanation, setting, exemplification, generalization, expectation
- *Metonymy:* replacing one object for another, as in "Chrysler announced..."

Probabilistic language models

- Are distinguished from *logical* models (grammars) and enable
  - processing of text in which speakers use different versions of language
  - disambiguation by choosing likely meaning
- Can generate *bigram* (2-word), *unigram*, and *trigram* probabilistic models from large text bases



Information extraction

- *Definition:* creation of database or knowledge base entries from text
- *Example:* "17 in XVGA Monitor for only $249.99" generates $(\exists m)$ $(m \in \text{Monitors} \wedge \text{Size}(m, \text{Inches}(17)) \wedge \text{Price}(m, 249.99))$
- *Regular expressions* can help analysis of text for attribute information
- Extraction works well in restricted domains when subjects of discourse are known

Machine translation

- Translation from source to target language
- *Types*
  - Rough
  - Restricted source (limited subj. matter, format)
  - Pre-edited source
  - Literary (beyond current software capabilities)
- *Challenge:* different languages have expressions with overlapping meanings, lacking the 1-to-1 correspondence convenient for translating

Subtopic outcome

5.5 Explain concepts of *natural-language processing*

References

George Luger. *Artificial Intelligence*. Addison Wesley, 2005.

Tom Mitchell. *Machine learning*. McGraw-Hill, 1997.

Stuart Russell and Peter Norvig. *AI: A Modern Approach, 2nd ed.*, Prentice Hall, 2003.

Paul Thagard. *Mind: Introduction to Cognitive Science*, 2nd ed. MIT press, 2005.