Review of Knowledge Representation

Review: Knowledge Representation (KR)

Questions Bob posed for any KR scheme:

1. *What* “language” is being used?
2. *What* knowledge is represented explicitly by a given set of expressions in the language?
3. *What* inference mechanism is available to make explicit knowledge which is implicit in the given set of expressions?

Questions Laurie and Chris added to the discussion:

4. *How* is the knowledge represented? *(in the brain)*
5. *How* is inference done? *(in the brain)*

Review (cont’d): Knowledge Representation (KR)

Chris quote, October 28, 2014,

> “COGS stumbles over what we mean by *model*”

1. Competence versus Performance
2. Description versus Explanation
3. Psychological Reality
4. Marr’s three levels (September 11, 2014, lecture)

Knowledge Representation in Machine Learning

1. Knowledge is represented in the form of probability distributions
2. Inference is done by numerical computation (to combine/update probability distributions into new probability distributions)
3. Bayes’ theorem plays a central role in statistical inference

Bayes’ Theorem (aka Bayes’ Rule)

Let H be the hypothesis (i.e., belief) and let E be the evidence

$$\underset{\text{posterior probability}}{\underset{\uparrow}{P(H \mid E)}} = \overset{\text{prior probability}}{\overset{\downarrow}{P(H)}} \frac{\overset{\text{likelihood}}{\overset{\downarrow}{P(E \mid H)}}}{\underset{\substack{\text{unconditional probability}\\ \text{(marginal likelihood)}}}{\underset{\uparrow}{P(E)}}}$$

Assignment 3 Question 8

Let H be the hypothesis that the student originally transferred from the Wednesday lab is male

Let E be the evidence that the student subsequently selected from the Monday lab is male

P(H) is calculated based on information stated in the problem

P(H|E) is calculated using Bayes' theorem

Interpretation: Evidence, E, updates the probability of the hypothesis, H, from P(H) to P(H|E)

"Big Data"

Corpora (from which probability distributions can be estimated)

Approach	*Example*
Structured	Cyc NTT's ALT-JE
Semi-structured	Wikipedia
Unstructured	Google (the entire WWW)

Representation of Linguistic Knowledge

Two "ways" to represent linguistic knowledge

1. Generative approach (linguistic knowledge *in the mind*)
   - There is a grammar
   - Language is the set of strings generated by the grammar
   - A speaker's knowledge of language consists of that grammar

2. Exemplar approach (linguistic knowledge *from input*)
    - Creates utterances by re-using pieces (of any size) previously recorded

Generative view: Representation of linguistic knowledge is **NOT** a surrogate (for world knowledge). You **DO** have language in your head. What is in your head **IS** language

Representation of Linguistic Knowledge

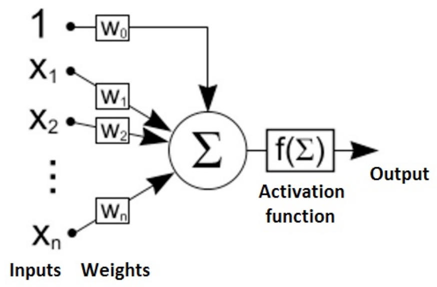
Generative Grammar

- Phrase structure, transformations
- Smallest, independent
- Phrase structure rules, transformations
- Generate linguistic experience
- Corpus: intuitive judgments
- "Goodness" = intuition

Exemplar Theory

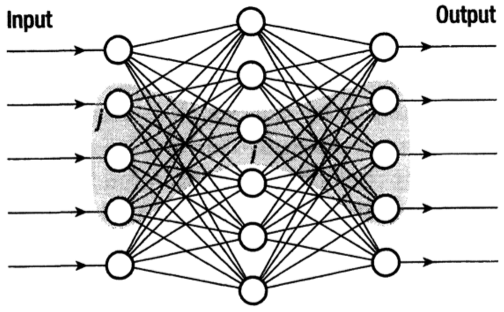
- Well-formed linguistic representation
- Subtrees, any size
- De-composing, composing operations
- Record/re-use all linguistic experience
- Corpus: linguistic input
- "Goodness" = probability

Connectionism: McCulloch–Pitts Neuron



Aside: As a (potential) hardware computing element, the McCulloch–Pitts neuron is "universal" in that it can be used to implement Boolean operations AND, OR and NOT

Connectionism: Neural Network



This example is a fully connected, three layer, feed forward neural network with five input nodes, five output nodes and six middle layer (i.e., hidden) nodes

In this example, there are a total of $5 \times 6 = 30$ connections between the input layer and the middle layer and a total of $6 \times 5 = 30$ connections between the middle layer and the output layer for a grand total of $30 + 30 = 60$ connections. Since there is a weight associated with each connection, there are a total of 60 weights in this network. Said another way, the behaviour of the neural network in this example is fully specified by the 60 weights involved. Said a third way, programming this neural network consists entirely and exclusively of specifying the 60 real numbers to be used as the weights.

If there are no further restrictions placed on the weights then the network, in fact, has infinite memory. Each of the 60 weights potentially is the memory equivalent of the infinite tape of a Turing machine (since the string of zeros and ones required to represent an arbitrary real number is unbounded in length). Suppose, instead, the real numbers were represented in standard IEEE single precision format. In this format, each real number uses 4 bytes (32 bits) of memory. Thus, the 60 weights use $60 \times 4 = 240$ bytes of memory, less than the number of characters in two “tweets.” If standard IEEE double precision format is used, which is the norm for scientific computation, then each real number uses 8 bytes (64 bits) of memory and the weights use a total of 480 bytes of memory, less than the number of characters in four “tweets.”

A fully connected, three layer feed forward architecture does not scale well with the number of nodes in each layer. Suppose the number of nodes at each layer is n. Then, by the argument given in the first paragraph of this section, a grand total of $2n^2$ connections/weights are required. Said another way, in this architecture, the number of weights required grows as the square of the number of input/output nodes.

Chris mentioned a specific example from Plunkett and Marchman, 1993. In that example, there were eighteen input units, twenty output units and thirty hidden units. Assuming full connectivity, there would be at total of $18 \times 30 + 30 \times 20 = 1,140$ connections/weights in the network.

Features of Connectionist Representation

1. *Distributed:* Each of the representations generated is composed from several computationally independent parts of the system
2. *Overlapping:* The different representations generated are composed from overlapping parts
3. *Holographic:* Any part of any given representation carries information about every part of the content represented by it
4. *Learning:* If a given neural network can perform a task then it can be trained to perform that task automatically (from examples)

Challenges to Connectionist Representation

1. There is no “universal” connectionist architecture. That is, there is no neural network that is capable of simulating the behaviour of any other neural network
2. For an arbitrary Turing computable task, one can not determine if there exists a neural network capable of performing that task
3. The number of training examples and the number of iterations required for back propagation to converge do not scale well with the size (i.e., with the number of nodes) of a fully connected neural network, even for tasks that the network can perform
4. Successful deployment of neural networks remains an art, involving restrictions on the number of nodes in one or more of the hidden layers and restrictions on the degree of connectivity between nodes in one layer to the next