Review

- Constraint satisfaction
  - Units are hypotheses about parts of solutions; weights code constraints between hypotheses
  - Network settles to pattern that maximizes "goodness" (satisfying constraints)
  - Supports content-addressable memory ("find a Shark in his 20s...")

- Correlational learning (Hebb rule)
  - Works perfectly only if input patterns are orthogonal (no similarity)
  - Response is combination of all trained patterns, weighted by similarity to current input

- Error-correcting learning (Delta rule)
  - Works perfectly if input patterns are linearly independent (something unique about each)
  - Guaranteed to find weights that produce zero error (if they exist)
  - Cannot even categorize correctly if assignment of targets to inputs is linearly inseparable (e.g., XOR), where similar inputs tend to correspond to different outputs

- Localist vs. distributed representations
  - IAC model of letter and word recognition; TRACE model of speech recognition
  - Representation is localist (1-to-1) or distributed (many-to-many) only relative to set of items
  - Distributed representations can learn both general and specific/idiosyncratic knowledge

Review (2)

- Back-propagation (generalization of Delta rule to multilayer networks)
  - Can learn linearly inseparable problems by re-representing inputs over intermediate ("hidden") units whose patterns have different similarities

- Generalization and overfitting
  - Add cost measure to penalize "complexity" of network (e.g., weight decay, to keep it more linear); improves generalization by avoiding overfitting training data
  - Cross-validation estimates overfitting based on subset of training data ("validation" set)

- Temporal learning (fully and simple recurrent networks)
  - Back-propagation can be applied to recurrent networks if past unit activations are stored (back-propagation through time)
  - Simple recurrent networks (SRNs) make previous hidden activations available as an additional (context) input; computationally efficient way to learn sequential tasks (e.g., prediction)

- Boltzmann machines; contrastive Hebbian learning
  - Contrastive Hebbian learning generalizes the Hebb rule for multilayer network (with symmetric weights; intrinsically settles to stable "attractor")
  - Mean-field version (Deterministic Boltzmann machines) is comparable to back-propagation

Review (3)

- Unsupervised learning; forward models
  - Unsupervised: Learn only from statistical distribution of inputs (e.g., generative models)
  - Auto-encoders and SRNs for sequential prediction are "self-supervised" (inputs = targets)
  - Forward model: Learned internal model of world that can convert distal error (due to outcomes) into proximal error (due to actions)

- Deep learning
  - Many hidden layers between inputs and outputs; can learn complex, hierarchical features sets
  - Practical at realistic scale due to technical advances, computational power, and massive data
  - Network and training data are structured to promote needed types of generalization

- Reinforcement learning
  - Environment provides evaluative feedback; not specification of correct action in each situation (supervised)
  - Temporal difference learning: Maximize expected future (discounted) reward
  - Can be combined with deep learning in complex, continuous environments

Designing input and output representations

- Constrain by empirical data when possible
- Localist representations
  - Useful for enforcing independence
  - Won’t support generalization
- Distributed representations
  - Pattern overlap should capture relevant similarity structure
  - Degree of sharing of features across entire training set matters
- Binary features: code as binary states (0/1)
- Real-valued features: discretize into separate “bins” (?)
  - Drawback: loses similarity of values in neighboring bins (coded by non-overlapping units)
How to represent real-valued features/properties?

- Map to single real-valued unit activation?
  - Because (input) activations are multiplied by outgoing weights, activations near 0 have less direct impact on the network than activations near 1
- Map to two opposing real values (activations summing to 1)
  - No 0/1 asymmetry (all inputs have same total activation)
  - Difficult to treat different ranges of values differently/independently
- Code value over bank of units (each “centered” on particular value)
  - Unit activations are Gaussian function of difference between unit’s “preferred” value and coded value
  - Can also use simple linear interpolation