Overview

- More on backprop
- Self-organizing maps

Another Application of Backpropagation: Image Compression

- Image compression
  1. target output is the same as the input.
  2. hidden layer units are fewer than the output (and input) layer units.
  3. the hidden layer forms the compressed representation.

Improving Backpropagation

To overcome the local minima problem:

- Adding momentum
  \[ \Delta W_{j,i}(t) = \alpha \times \Delta_i \times I_j + \eta \times \Delta W_{j,i}(t - 1) \]

- Incremental update (as opposed to batch update) with random input-target order.

- Add a little bit of noise to the input.

- Allow increasing \( E \) with a small probability, as in Simulated Annealing.

Backpropagation Exercise

- URL: http://www.cs.tamu.edu/faculty/choe/src/backprop-1.6.tar.gz

- Untar and read the README file:
  \[
  \text{gzip} -dc \text{backprop-1.6.tar.gz} | \text{tar xvf -}
  \]

- Run make to build (on departmental unix machines).

- Run ./bp conf/xor.conf etc.

**Backpropagation: Example Results**

- Epoch: one full cycle of training through all training input patterns.
- OR was easiest, AND the next, and XOR was the most difficult to learn.
- Network had 2 input, 2 hidden and 1 output unit. Learning rate was 0.001.

**Backpropagation: Things to Try**

- How does increasing the number of hidden layer units affect the (1) time and the (2) number of epochs of training?
- How does increasing or decreasing the learning rate affect the rate of convergence?
- How does changing the slope of the sigmoid affect the rate of convergence?
- Different problem domains: handwriting recognition, etc.

**Unsupervised Learning**

- No teacher signal (i.e. no feedback from the environment).
- The network must discover patterns, features, regularities, correlations, or categories in the input data and code them in the output.
- The units and connections must display some degree of self-organization.
- Unsupervised learning can be useful when there is redundancy in the input data.
- A data channel where the input data content is less than the channel capacity, there is redundancy.
What Can Unsupervised Learning Do?

- **Familiarity**: how similar is the current input to past inputs?
- **Principal Component Analysis**: find orthogonal basis vectors (or axes) against which to project high dimensional data.
- **Clustering**: \( n \) output class, each representing a distinct category. Each cluster of similar or nearby patterns will be classified as a single class.
- **Prototyping**: For a given input, the most similar output class (or exemplar) is determined.
- **Encoding**: application of clustering/prototyping.
- **Feature Mapping**: topographic mapping of input space onto output network configuration.

**SOM Algorithm**

1. Randomly initialize reference vectors \( w_i \)
2. Randomly sample input vector \( x \)
3. Find Best Matching Unit (BMU):
   \[
i(x) = \arg\min_j \| x - w_j \|
   \]
4. Update reference vectors:
   \[
w_j \leftarrow w_j + \alpha \Lambda(j, i(x))(x - w_j)
   \]
   \( \alpha \): learning rate
   \( \Lambda(j, i(x)) \): neighborhood function of BMU.
5. Repeat steps 2 – 4.

**Typical Neighborhood Functions**

- Gaussian: \( \Lambda(j, i(x)) = \exp\left(-\|r_j - r_i(x)\|^2/2\sigma^2\right) \)
- Flat: \( \Lambda(j, i(x)) = 1 \) if \( |r_j - r_i(x)| \leq \sigma \), and 0 otherwise.
- \( \sigma \) is called the neighborhood radius.
**Training Tips**

- **Start with large neighborhood radius.**
- Gradually decrease radius to a small value.
- *Star* t with large neighborhood radius.
- Gradually decrease radius to a small value.

**Properties of SOM**

- Edges: connect immediate neighbors on the map.
- Nodes: reference vectors (x and y coordinates).
- Each input is a point in Cartesian plane.
- Train with uniformly random 2D inputs.
- Properly connected immediate neighbors.
- Gradually decrease radius to a small value.
- Gradually decrease learning rate ρ.

**Example: 2D Input / 2D Output**

- More units represent input space that are more frequently sampled.
- Neighboring units represent nearby points.
- Topology preservation.
- Create output space.
- Maps continuous input space to discrete.
- Approximation of input space.

**Performance Measures**

- **Quantization Error**
  \[ u(x) = \begin{cases} 
  0 & \text{if the 1st and 2nd BMUs are not adjacent} \\
  1 & \text{otherwise} 
\end{cases} \]

- **Topographic Error**
  \[ (f(x) - f(y))^2 \sum_{i=1}^{N} \frac{1}{N} = \epsilon \]

- Average distance between each data vector and its BMU.
Different 2D Input Distributions

- What would the resulting SOM map look like?
- Why would it look like that?

High-Dimensional Inputs

- SOM can be trained with inputs of arbitrary dimension.
  - Dimensionality reduction: N-D to 2-D.
  - Extracts topological features.
  - Used for visualization of data.

Applications

- Data clustering and visualization.
- Optimization problems:
  - Traveling salesman problem.
- Semantic maps:
  - Natural language processing.
- Preprocessing for signal and image-processing.
  2. Phonetic map for speech recognition.

Exercise

1. What happens when \( N_i(x) \) and \( \alpha \) was reduced quickly vs. slowly?
2. How would the map organize if different input distributions are given?
3. For a fixed number of input vectors from real-world data, a different visualization scheme is required. How would you use the number of input vectors that best match each unit to visualize the property of the map?
Key Points

- How can backprop be improved?
- What are the various ways to apply backprop?
- SOM basic algorithm
- What kind of tasks is SOM good for?

Next Time

- Recurrent networks
- Chapter 20, section 20.8: Genetic algorithms