Overview

- SOM demo
- Recurrent networks
- Genetic Algorithms

SOM Example: Handwritten Digit Recognition

- Preprocessing for feedforward networks (supervised learning).
- Better representation for training.
- Better generalization.

SOM Demo

Jochen Fröhlich’s *Neural Networks with JAVA* page:
http://rfhs8012.fh-regensburg.de/saj39122/jfroehl/diplom/e-index.html

Check out the Sample Applet link.

SOM Demo: Traveling Salesman Problem

Using Fröhlich’s SOM applet:
- 1D SOM map ($1 \times n$, where $n$ is the number of nodes).
- 2D input space.
- Initial neighborhood radius of 8.
- Stop when radius $< 0.001$.
- Try 50 nodes, 20 input points.

Click on [Parameters] to bring up the config panel. After the parameters are set, click on [Reset] in the main applet, and then [Start learning].
SOM Demo: Space Filling in 2D

Using Fröhlich’s SOM applet:

- 1D SOM map ($1 \times n$, where $n$ is the number of nodes).
- 2D input space.
- Initial neighborhood radius of 100.
- Stop when radius $< 0.001$.
- Try 1000 nodes, and 1000 input points.

SOM Demo: Space Filling in 3D

Using Fröhlich’s SOM applet:

- 2D SOM map ($n \times n$, where $n$ is the number of nodes).
- 2D input space.
- Initial neighborhood radius of 10.
- Stop when radius $< 0.001$.
- Try $30 \times 30$ nodes, and 500 input points. Limit the $y$ range to 15.
  Also try $50 \times 50$, 1000 input points, and 16 initial radius.

Recurrent Networks

Connection graph can contain cycles, e.g. reciprocal connections: i.e. not strictly feed-forward.

- Statistical mechanics based models (associative or content-addressable memory): Hopfield network, Boltzman machines, etc.
- Sequence encoding: Simple Recurrent Network, etc.
- Other biologically motivated networks: laterally connected self-organizing maps, etc.

Simple Recurrent Network (Elman Network)

- Sequence encoding.
- Hidden layer activation from previous time step used as input.
- Use standard back-propagation learning.
Example input and target sequence: output 1 when two 1s appear in a row.

Time: 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19
Input: 1 0 1 0 1 1 0 0 0 1 1 1 1 0 1 0 1 1 0 1
Target: 0 0 0 0 0 1 0 0 0 0 1 1 1 0 0 0 0 1 0 0

Evolution as a problem solving strategy:
- population of solutions, where each gene represents an individual
- selection based on fitness function: survival of the fittest
- mating (cross-over) and reproduction
- random mutation

Properties of Genetic Algorithms
- each gene encodes a solution
- similar to hill-climbing search
- parallel search
- works for both immediate or delayed reward

Designing a GA Solution to a Problem
There are many different issues:
- What is the fitness function?
- How is an individual gene represented (how to encode) and what does it represent?
- How are individuals selected?
- How do individuals reproduce?
Issues in GA: Diversity

How to maintain diversity:

- Letting only successful ones to reproduce can seriously reduce the gene pool and an epidemic can wipe out the whole population: solution cannot generalize in new and unexpected conditions.
- Converged population can often times be found at local minima, not at the global optimum.

More Issues in GA

- Cross-over strategy: success depends on how genes are encoded (or represented).
- Not too much theoretical understanding about why it works so well.
- Crevices in fitness landscape: similar to spikes in hill climbing.
- How to combine learning with evolution.
- How to use cultural leverage.

GA as a Learning Algorithm

- An individual gene may not seem to learn, but when we look at the evolution of individuals over time, they can be seen as adapting, and thus learning to cope with the environment.
- If each individual encodes a function rather than a simple solution, the above point becomes clearer. At each generation, the parameters in the function can be seen as being adapted.
- Fitness can then be measured by using the function with the given parameters in specific tasks.

GA as a Learning Algorithm: Neuroevolution

Evolving neural networks:

- Genes encode neural networks (connection topology and connection weights).
- Evaluate, select, and reproduce new population of neural networks.

Problem: individual neurons performing good work may get lost.
Neuroevolution: Evolving Individual Neurons

- Genes encode individual neurons.
- Neurons solve sub-problems and the ones that solve the problem well get a chance to participate in a network in the next generation.
- Better diversity is maintained.

Key Points

- SOM: Try out the effects of different parameters, network size, 1D or 2D map, neighborhood radius, etc.
- Simple recurrent networks: how can it encode sequences, how is it different from standard backprop and who similar is it?
- Genetic algorithms basics.
- What are the issues to be solved in genetic algorithms.

Next Time

- Decision Tree Learning

GA Demo

- Generation of melodies (Chen and Miikkulainen)
- Gaming AI; harvesters and predators (Stanley and Miikkulainen)
- Non-markovian control (Gomez and Miikkulainen)