Evolving Neural Networks

This lecture is based on Xin Yao's tutorial slides
"*From Evolving Single Neural Networks to Evolving Ensembles*"
presented at CEC 2007.

Literature:

Evolving Artificial Neural Networks, Xin Yao,
Proceedings of the IEEE,Volume 87, Issue 9, Sep 1999
Page(s):1423 - 1447
http://www.cs.bham.ac.uk/~xin/papers/published_iproc_sep99.pdf

Good Internet article:
Evolutionary Neural Networks: Design Methodologies

Evolving Neural Networks (cont.)

- Learning and evolution are two fundamental forms of adaptation
- Simulated evolution can be introduced into neural networks at different levels

Idea: integration of neural and evolutionary computation

- for evolving neural network connection weights
- for evolving neural network architectures
- for evolving neural network learning rules

What Is Evolutionary Computation

- It is the study of computational systems which use ideas and get inspirations from natural evolution
- One of the principles borrowed is *survival of the fittest*
- Evolutionary computation techniques can be used in optimization, learning and design
- Evolutionary computation techniques do not require rich domain knowledge to use. However, domain knowledge can be incorporated into evolutionary computation techniques

A Simple Evolutionary Algorithm (EA)

1. Generate the initial population $P(0)$ at random, and set $i = 0$
2. REPEAT
   (a) Evaluate the fitness of each individual in $P(i)$
   (b) Select parents from $P(i)$ based on their fitness in $P(i)$
   (c) Generate offspring from the parents using *crossover* and *mutation* to form $P(i+1)$
   (d) $i = i + 1$
3. UNTIL halting criteria are satisfied

Evolving Connection Weights

- Recall: supervised learning in neural networks was formulated as minimization of an error function (sum-of-squares error, cross-entropy error)
- Gradient-based optimization techniques are often used, but
  - they require the error function to be differentiable
  - they are sensitive to initial conditions
  - they get trapped in local minima
  - they can converge slowly (if no acceleration tricks are used)
- EAs are good at dealing with complex and nondifferentiable functions. They are robust and less sensitive to initial conditions

Encoding Connection Weights

Before evolution, and encoding scheme (genotypic representation) is often needed to represent neural networks. There are different methods for representing weights.

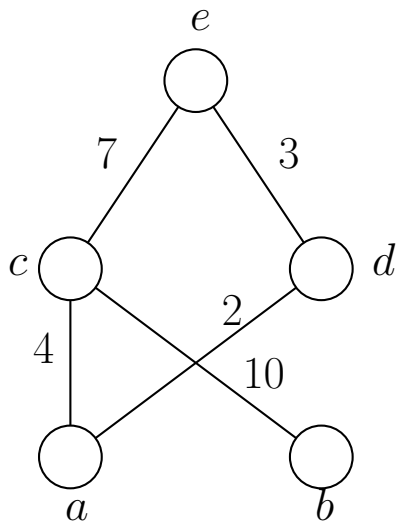
- Binary representation
- Real number representation
- Hybrid method

The Evolution of Connection Weights

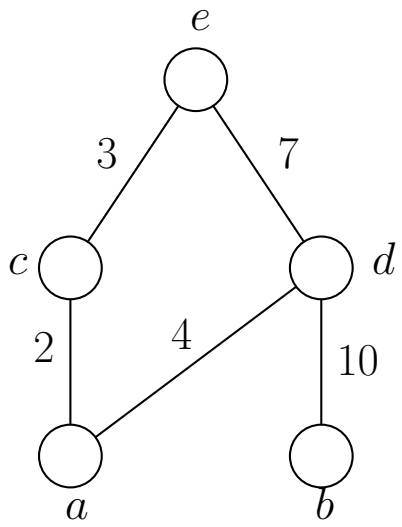
Assumption: the neural network architecture is fixed during the evolution

1. Decode each individual (genotype) into a set of connection weights (phenotype)
2. Evaluate the fitness (error) of each individual
3. Reproduce a number of children for each individual in the current generation according to its fitness
4. Apply genetic operators, such as crossover and mutation, to each child individual generated above and obtain the next generation

Permutation Problem



	a	b	c	d	e
a	0000	0000	0100	0010	0000
b	0000	0000	1010	0000	0000
c	0000	0000	0000	0000	0111
d	0000	0000	0000	0000	0011
e	0000	0000	0000	0000	0000



	a	b	c	d	e
a	0000	0000	0010	0100	0000
b	0000	0000	0000	1010	0000
c	0000	0000	0000	0000	0011
d	0000	0000	0000	0000	0111
e	0000	0000	0000	0000	0000

Caused by the many to one mapping from genotypes to phenotypes.

Discussions on Evolutionary Training

- Evolutionary training is attractive because it can handle the complex and nondifferentiable space better. It is particularly appealing when gradient information is unavailable or very costly to obtain or estimate
- Evolutionary training may be slow for some problems in comparison with fast gradient descent algorithms. However, it always searches for a globally optimal solution.
- For some problems, evolutionary training is significantly faster and more reliable than gradient descent algorithms

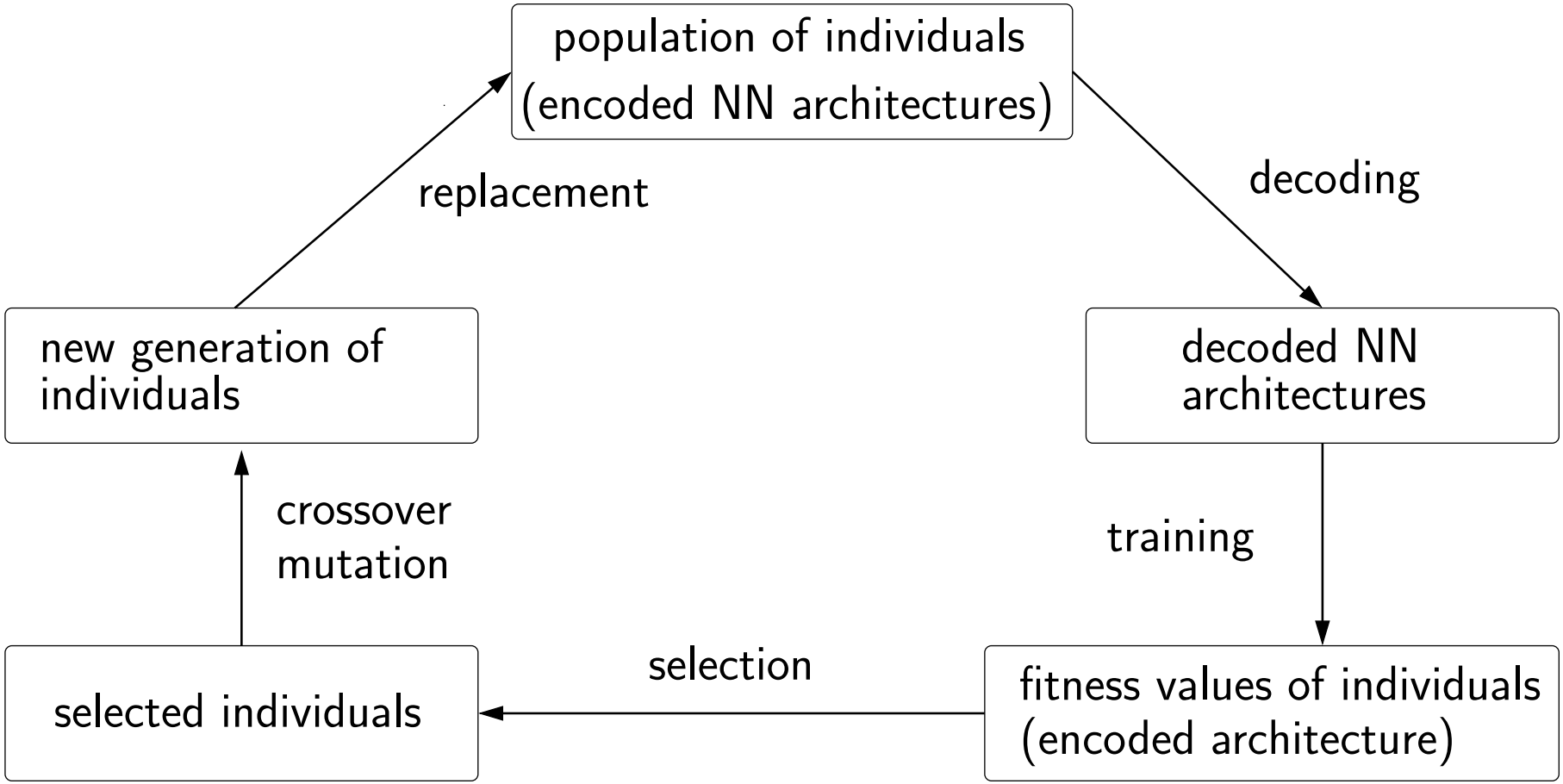
Hybrid Training

- Global search techniques are always in conflict between exploitation and exploration
- EAs are not very good at fine-tuned local search although they are good at global search
- Efficiency of evolutionary training can be improved significantly by incorporating a local search procedure into the evolution, i.e., combining EA's global search ability with local search's fine-tuning ability
- Other alternative: use EAs to search for a near optimal set of connection weights and then use a local search algorithm to fine tune the weights

Evolving NN Architectures

- Neural networks have mostly been designed manually
- Design of a near optimal neural network architecture can be formulated as a search problem in the architecture space
- Evolutionary algorithms are good candidates for searching this space

Evolving NN Architectures: Common Practice



Typical Cycle of Evolving NN Architectures

1. Decode each individual in the current generation into an architecture
2. Train each neural network with the decoded architecture starting from different sets of random initial connection weights
3. Calculate the fitness of each individual
4. Reproduce a number of children for each individual in the current generation based on its fitness
5. Apply variation operators to the children generated above and obtain next generation

Fitness Evaluation

Fitness is a measure of the quality of the found solution.
Fitness measure can based on

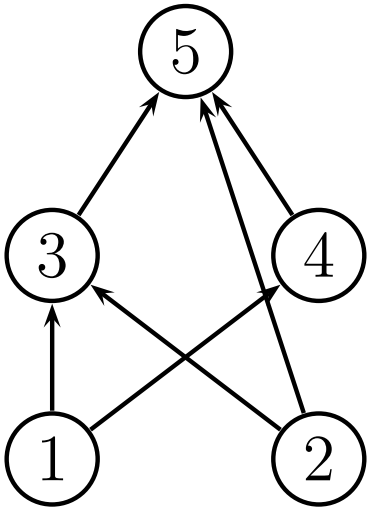
- Training error
- Testing error
- Training/testing error and network complexity
  - based on the number of connections
  - based on some information criterion, such as AIC or MDL

Direct Encoding Scheme

- In the direct encoding scheme, each connection is an architecture is directly specified by its binary representation
- An $N \times N$ matrix $\mathbf{C}$ can represent an architecture with N units, where $\mathbf{C}_{ij}$ indicates presence or absence of the connection from unit i to unit j
- Each such matrix has a direct one-to-one mapping to the corresponding architecture. The binary string representing an architecture is just the concatenation of rows (or columns) of the matrix.

Feedforward Network Encoding (Example)

$$\begin{pmatrix} 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$



$$0110 \quad 101 \quad 01 \quad 1$$

- Direct encoding scheme is simple and straightforward to implement
- Found solutions are tight and contains interesting designs that no one has hit upon so far
- It does not scale well for large architectures