Today’s Main Topic

• Neuroevolution: Evolve artificial neural networks to control behavior of robots and agents.
• Main idea: Mimic the natural process of evolution that gave rise to the brain, the source of intelligence.
  – Population
  – Competition
  – Selection
  – Reproduction and mutation

Why Neuroevolution?

• Neural networks already successful in many domains.
• However, in certain domains, it is hard to fit the existing framework and learning algorithms.
• Hard domains: fin-less rocket control, robotic agent control, etc.
Outline

- Basic neuroevolution techniques
- Advanced techniques
  - E.g. combining learning and evolution
- Extensions to applications
- Application examples
  - Control, Robotics, Artificial Life, Games

Neuroevolution Basics

- A single chromosome encodes a full neural network.
- Each gene, a single bit (or a real number), maps to a connection weight in the neural network.

Neuroevolution Decision Strategies

- Input variables describe the state
- Output variables describe actions
- Network between input and output
  - Hidden nodes
  - Weighted connections
- Execution:
  - Numerical activation of input
  - Nonlinear weighted sums
- Performs a nonlinear mapping
  - Memory in recurrent connections
- Connection weights and structure evolved

Neuroevolution Basics: Operations

- Cross-over.
- Mutation.
Neuroevolution Basics: Cross-Over in Detail

- Cross-over of two individuals produces two offsprings with a mixed heritage.

Conventional Neuroevolution (CNE)

- Evolving connection weights in a population of networks
- Chromosomes are strings of weights (bits or real) – E.g. 10010110101100101111001 – Usually fully connected, fixed topology – Initially random

Problems with CNE

- Evolution converges the population (as usual with EAs) – Diversity is lost; progress stagnates
- Competing conventions – Different, incompatible encodings for the same solution
- Too many parameters to be optimized simultaneously – Thousands of weight values at once
**Advanced NE 1: Evolving Neurons**

- Evolving individual neurons to cooperate in networks \(^1,22,24\) (Agogino GECCO’05)
- E.g. Enforced Sub-Populations (ESP?)
  - Each (hidden) neuron in a separate subpopulation
  - Fully connected; weights of each neuron evolved
  - Populations learn compatible subtasks

**Advanced NE 2: Evol. Subpopulations**

- Evolution encourages diversity automatically
  - Good networks require different kinds of neurons
- Evolution discourages competing conventions
  - Neurons optimized for compatible roles
- Large search space divided into subtasks
  - Optimize compatible neurons

**Advanced NE 3: Evolving Topologies**

- Optimizing connection weights and network topology \(^11,40\)
- E.g. Neuroevolution of Augmenting Topologies (NEAT \(^27,29\))
- Based on *Complexification*

**How Can We Complexify?**

- Can optimize not just weights but also topologies
- Solution: Start with minimal structure and complexify \(^37\)
- Can search a very large space of configurations!

---

**Advanced NE 2: Evol. Subpopulations**

- Evolution encourages diversity automatically
  - Good networks require different kinds of neurons
- Evolution discourages competing conventions
  - Neurons optimized for compatible roles
- Large search space divided into subtasks
  - Optimize compatible neurons

**Advanced NE 3: Evolving Topologies**

- Optimizing connection weights and network topology \(^11,40\)
- E.g. Neuroevolution of Augmenting Topologies (NEAT \(^27,29\))
- Based on *Complexification*

**How Can We Complexify?**

- Can optimize not just weights but also topologies
- Solution: Start with minimal structure and complexify \(^37\)
- Can search a very large space of configurations!
How Can Crossover be Implemented?

- Problem: Structures do not match

![Diagram of crossover problem](image)

- Solution: Utilize historical markings

<table>
<thead>
<tr>
<th>Genome (Genotype)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node 1</td>
</tr>
<tr>
<td>Node 2</td>
</tr>
<tr>
<td>Node 3</td>
</tr>
<tr>
<td>Node 4</td>
</tr>
<tr>
<td>Node 5</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Constant</th>
<th>Genes</th>
</tr>
</thead>
<tbody>
<tr>
<td>In 1</td>
<td>Out 4</td>
</tr>
<tr>
<td>In 2</td>
<td>Out 4</td>
</tr>
<tr>
<td>In 3</td>
<td>Out 5</td>
</tr>
<tr>
<td>In 4</td>
<td>Out 5</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Network (Phenotype)</th>
</tr>
</thead>
<tbody>
<tr>
<td>In 1</td>
</tr>
<tr>
<td>In 2</td>
</tr>
<tr>
<td>In 3</td>
</tr>
</tbody>
</table>

Further Neuroevolution Techniques

- Incremental evolution\(^{13,33,39}\)
- Utilizing population culture\(^{2,18}\)
- Evolving ensembles of NNs\(^{16,23,36}\) (Pardoe GECCO’05)
- Evolving neural modules\(^{25}\)
- Evolving transfer functions and learning rules\(^{4,26}\)
- Combining learning and evolution

How can Innovation Survive?

- Problem: Innovations have initially low fitness

![Diagram of innovation survival](image)

- Solution: Speciate the population
  - Innovations have time to optimize
  - Mitigates competing conventions
  - Promotes diversity

Further Neuroevolution Applications

- Evolving composite decision makers\(^{36}\)
- Evolving teams of agents\(^{3,28,41}\)
- Utilizing coevolution\(^{30}\)
- Real-time neuroevolution\(^{28}\)
- Combining human knowledge with evolution\(^{8}\)
Applications to Control

- Pole-balancing benchmark
  - Originates from the 1960s
  - Original 1-pole version too easy
  - Several extensions: acrobat, jointed, 2-pole, particle chasing

- Good surrogate for other control tasks
  - Vehicles and other physical devices
  - Process control

Competitive Coevolution with NEAT

- Complexification elaborates instead of alters
  - Adding more complexity to existing behaviors

- Can establish a coevolutionary arms race
  - Two populations continually outdo each other
  - Absolute progress, not just tricks

Competitive Coevolution

- Evolution requires an opponent to beat
- Such opponents are not always available
- Co-evolve two populations to outdo each other
- How to maintain an arms race?

Robot Duel Domain

- Two Khepera-like robots forage, pursue, evade
  - Collect food to gain energy
  - Win by crashing to a weaker robot
Early Strategies

• Crash when higher energy
• Collect food by accident
• DEMO

Mature Strategies

• Collect food to gain energy
• Avoid moving to lose energy
• Standoff: Difficult to predict outcome
• DEMO

Sophisticated Strategy

• “Fake” a move up, force away from last piece
• Win by making a dash to last piece
• Complexification → arms race
• DEMO

Applications to Games

• Good research platform
  – Controlled domains, clear performance, safe
  – Economically important; training games possible

• Board games: beyond limits of search
  – Evaluation functions in checkers, chess
  – Filtering information in go, othello
Discovering Novel Strategies in Othello

- Players take turns placing pieces
- Each move must flank opponent's piece
- Surrounded pieces are flipped
- Player with most pieces wins

Evolving Against a Random Player

- Network sees the board, suggests moves by ranking
- Networks maximize piece counts throughout the game
- A positional strategy emerges
- Achieved 97% winning percentage

Evolving Against an $\alpha$-$\beta$ Program

- Iago’s positional strategy destroyed networks at first
- Evolution turned low piece count into an advantage
- Mobility strategy emerged!
- Achieved 70% winning percentage
**Example game**

- Black’s positions strong, but mobility weak
- White (the network) moves to f2
- Black’s available moves b2, g2, and g7 each will surrender a corner
- The network wins by forcing a bad move

**Discovering Novel Strategies**

- Neuroevolution discovered a strategy novel to us
- “Evolution works by tinkering”
  - So does neuroevolution
  - Initial disadvantage turns into novel advantage

**Future Challenge: Utilizing Knowledge**

- Given a problem, NE discovers a solution by exploring
  - Sometimes you already know (roughly) what works
  - Sometimes random initial behavior is not acceptable
- How can domain knowledge be utilized?
  - By incorporating rules (Yong GECCO’05)
  - By learning from examples

**Numerous Other Applications**

- Creating art, music
- Theorem proving
- Time-series prediction
- Computer system optimization
- Manufacturing optimization
- Process control optimization
- Etc.
Conclusion

- Neuroevolution, mimicking the natural process of evolution, is an effective strategy for constructing complex and useful behavior.
- Neuroevolution often performs well for reinforcement learning tasks.
- Analyzing the resulting network is a challenge.

References


