Comparing Direct and Indirect Encodings Using Both Raw and Hand-Designed Features in Tetris

By Lauren Gillespie, Gabby Gonzales and Jacob Schrum

gillespl@southwestern.edu  gonzale9@alumni.southwestern.edu  schrum2@southwestern.edu
Introduction

- Challenge: Use less domain-specific knowledge
  - Important for general game-playing agents
  - Requires using raw features
  - Difficult to train agents

- This Research
  - Compare evolutionary algorithm HyperNEAT to NEAT
  - See if indirect encoding of HyperNEAT advantageous
  - Also compare with hand-designed features
Tetris Domain

- Consists of 10 x 20 game board
- Orient tetrominoes to clear lines
- Clearing multiple lines = more points
- Hole: open spaces with at least one block above

Previous Work
- All use hand-designed features
- Reinforcement learning$^\dagger$ and evolutionary computation$^\ddagger$

### Hand-Designed vs. Raw Features

<table>
<thead>
<tr>
<th><strong>Hand-Designed</strong></th>
<th><strong>Raw</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>• Hand-picked information of game state as input</td>
<td>• One feature per game state element</td>
</tr>
<tr>
<td>• User processes input</td>
<td>• Little input processing</td>
</tr>
<tr>
<td><strong>Pros:</strong> ✷ Smaller input space, easier to learn</td>
<td><strong>Pros:</strong> ✷ Less limited by domain†</td>
</tr>
<tr>
<td><strong>Cons:</strong> ✷ Very domain-specific, not versatile</td>
<td>✷ Less human expertise needed</td>
</tr>
<tr>
<td>• Human expertise needed</td>
<td><strong>Cons:</strong> ✷ Large input space &amp; networks</td>
</tr>
<tr>
<td></td>
<td>• Harder to learn, more time</td>
</tr>
</tbody>
</table>
**NEAT vs. HyperNEAT**

Evolved network and agent network

**NEAT**

- Direct Encoding

**HyperNEAT**

- Indirect Encoding

**Evolved network (CPPN)**

**Agent network**

- **UTILITY**

- **Bias**

- **X**

- **Y**
Raw Features Setup

• Board configuration:
  ✦ 2 input sets: location of all blocks, location of all holes
• NEAT: Inputs sets given as linear sequence
• HyperNEAT: Two 2D input substrates
Hand-Designed Features Setup

- Bertsekas et al. features† plus additional hole per column feature
- All scaled to [0,1]
  - Column height
  - Height difference
  - Tallest column
  - Number of holes
  - Holes per column

† Bersekas et al. 1996. Neuro-Dynamic Programming

HyperNEAT Setup
NEAT vs. HyperNEAT: Raw Features

Game Score vs. Generation for HyperNEAT Raw and NEAT Raw.
Raw Features Champion Behavior

NEAT with Raw Features

HyperNEAT with Raw Features
Hand-Designed Features Behavior

NEAT with Hand-Designed Features

HyperNEAT with Hand-Designed Features
Conclusion

- Raw features
  - Indirect encoding of HyperNEAT effective
  - Geometric awareness an advantage
- Hand-designed features
  - Ultimately NEAT produced better agents
- Future work:
  - HybrID might combine strengths of both
  - Raw Features in other domains, Ms. Pac-Man
Questions?

• Contact info:
  gillespl@southwestern.edu
  schrum2@southwestern.edu
  gonzale9@alumni.southwestern.edu

• Movies and Code:
  https://tinyurl.com/tetris-gecco2017
Auxiliary Slides
Visualizing Substrates
Experimental Setup

- Agent networks evaluated each piece placement
- Each experiment evaluated with 30 runs
  - 500 generations/run, 50 agents/generation
  - Objectives averaged across 3 trials/agent
    - Noisy domain, multiple trials needed
- NSGA-II objectives: game score & survival time
NSGA-II

- Pareto-based multiobjective EA optimization
- Parent population, $\mu$, evaluated in domain
- Child population, $\lambda$, evolved from $\mu$ and evaluated
- $\mu + \lambda$ sorted into non-dominated Pareto fronts
  - **Pareto front**: All individual such that
    - $v = (v_1, \ldots, v_n)$ dominates vector $u = (u_1, \ldots, u_n)$ iff
      1. $\forall i \in \{1, \ldots, n\}: v_i \geq u_i$, and
      2. $\exists i \in \{1, \ldots, n\}: v_i > u_i$.
- New $\mu$ picked from highest fronts
- Tetris objectives: Game score, time
Visualizing Link Weights
Future Work

• HybrID†
  ✦ Start with HyperNEAT, switch to NEAT
  ✦ Gain advantage of both encodings

• Raw feature Tetris with Deep Learning

• Raw features in other visual domains
  ✦ Video games: DOOM, Mario, Ms. Pac-Man
  ✦ Board games: Othello, Checkers

NEAT

- NeuroEvolution of Augmenting Topologies†
- Synaptic and structural mutations
- Direct encoding
  - Network size proportional to genome size
- Crossover alignment via historical markings
- Inefficient with large input sets
  - Mutations do not alter behavior effectively

† Stanley & Miikkulainen. 2002. Evolving Neural Networks Through Augmenting Topologies
HyperNEAT

• Hypercube-based NEAT†
• Extension of NEAT
• Indirect encoding
  ✦ Evolved CPPNs encode larger substrate-based agent ANNs
• Compositional Pattern-Producing Networks (CPPNs)
  ✦ CPPN queried across substrate to create agent ANN
  ✦ Inputs = neuron coordinates, outputs = link weights
• Substrates
  ✦ Layers of neurons with geometric coordinates
  ✦ Substrate layout determined by domain/experimenter

† Stanley et al. 2009. A Hypercube-based Encoding for Evolving Large-scale Neural Networks
Evolutionary Algorithms (EA)

**NEAT**
- NeuroEvolution of Augmenting Topologies†
- Mutates structure and weights
- Direct encoding
  - Network size = genome size
- Inefficient with large input sets
  - Mutations not as effective

**HyperNEAT**
- Hypercube-based NEAT†
- Indirect encoding
  - Evolved CPPN *indirectly* plays game through agent network
- Geometric awareness
  - Agents can learn from domain geometry
- Better with large input sets
  - Geometric awareness gives agents more information

† Stanley & Miikkulainen. 2002. Evolving Neural Networks Through Augmenting Topologies
Afterstate Evaluation

- Evolved agents used as afterstate evaluators
- Determine next move from state after placing piece
- All possible piece locations determined, evaluated
- Placement with best evaluation from state chosen
- If placements lead to loss, not considered
- Agent moves piece to best placement, repeats
Raw Features Setup

- Board configuration:
  - Two input sets
    1. Location of all blocks
      - block = 1, no block = 0
    2. Location of all holes
      - hole = -1, no hole = 0
- NEAT: Inputs in linear sequence
- HyperNEAT: Two 2D input substrates