A Unified Approach to Evolving Plasticity and Neural Geometry

Kristiana Rendon, Luke Gehman, and Demitri Maestas
The Brain & Neuroevolution

Creating Artificial Neural Networks

- Hard to replicate brain as artificial neural networks (ANNs)
- Very dynamic, module, and regular
- Neuroevolution = autonomously generating ANNs
  - Evolutionary algorithms
  - Still can’t compare to real brain
  - Neural topology != neural topography
    - Important for spatial organization


NEAT
NeuroEvolution of Augmenting Topologies

- Evolves increasingly large ANNs
- Takes simple network → adds nodes/connections via mutations
- Searches networks
  - More complex network takes more time
- Direct encoding
  - Each part of solution (gene) gets its own mapping (BAD)
    - similar genes → different encoding → more searching
- Does not scale well
HyperNEAT

Hypercube-based NEAT

- **Indirect encoding**
  - Encode solution as function of geometry
    - patterns/regularities (symmetry, repetition)
  - Can compress and reuse these patterns
  - CPPNs

- **Nodes/connections need to be placed in certain geometric locations**
  - Exploit topography
  - Beneficial for neuroevolution
  - More like real brain
CPPNs

*Compositional Pattern Producing Networks*

- Abstracted version of DNA
  - Compactly encodes patterns of weights across network’s geometry
- Function input = node locations and role
- Function output = weights of connections
- Function return = topographic pattern (substrate)
- Composition of functions/regularities
  - Gaussian (symmetry) and periodic (repetition)
- Can be evolved by NEAT
1) Query each potential connection on substrate

2) Feed each coordinate pair into CPPN

3) Output is weight between \((x_i, y_i)\) and \((x_j, y_j)\)

HyperNEAT: Potential connections → CPPN → Weight of connections
Still Not Good Enough :(

- Static implementations
- No online adaptation
- Needs learning rules
- Needs to be more biologically plausible
- Needs to know locations and roles
- Evolvable-substrate and adaptive HyperNEAT can help
Evolvable Substrate HyperNEAT

- Locations of hidden nodes determined by CPPN
- The CPPN paints a picture of activations
- Chose nodes which give the most information using quadtree algorithm
Quadtree algorithm

Quadtree + band pruning
Adaptive HyperNEAT

- Want network which adapts to observations?
- CPPN produces parameters for Hebbian Learning

$$\Delta w_{ji} = \eta \cdot [A o_j o_i + B o_j + C o_i + D]$$
Evolved CPPN Topology
Adaptive ES-HyperNEAT

- Simultaneously evolves geometry, density, and plasticity, using a combination of the previously developed versions of NEAT.

- CPPN generates 6 additional outputs: Learning rate ($n$), Correlation term ($A$), presynaptic term ($B$), postsynaptic term ($C$), constant ($D$), and modulation ($M$). Used to simulate Hebbian learning!

\[
\Delta w_{ji} = \eta \cdot [A o_j o_i + B o_j + C o_i + D]
\]

\[
m_i = \sum_{w_{ji} \in Mod} w_{ji} \cdot o_j.
\]
Adaptive ES-HyperNEAT

- Each Neuron computes its own modulatory activation \( (m) \), which we use to adjust weights of connections between neurons

\[
m_i = \sum_{w_{ji} \in \text{Mod}} w_{ji} \cdot o_j.
\]

\[
\Delta w_{ji} = \tanh\left(\frac{m_i}{2}\right) \cdot \eta \cdot [A o_i + B o_j + C o_i + D].
\]

- Determines the placement and density of nodes from implicit information gained from the weight output and the modulatory output from the CPPN
Adaptive ES-HyperNEAT

An example of an ANN generated by it's respective CPPN
Continuous T-Maze Experiment

- Standard test of operant conditioning in animals
- Augmented T-Maze; Higher valued reward is achieved in sequence
- No sensor pre-processing needed, direct input into Adaptive ES-HyperNeat, sensors are correlated geometrically
- Fitness function is maximized when the same reward is consistently collected.
- Ran with:
  1000 generations, 300 individuals, 10% elitism
  Crossover offspring with no mutation (~50%) / direct offspring with mutation (~94%)
Results

- ES-HyperNEAT solving T-Maze at 1 out of 30 runs on average
- Adaptive ES-HyperNEAT found a solution in 19 out of 30 runs on average.
- Augmenting ES-HyperNEAT to adapt is important for adaptation tasks.
- No special sensors, only raw sensor input.
- Neural dynamics start to represent dynamics in nature.
- A single compact CPPN can encode a full adaptive network with full plasticity.