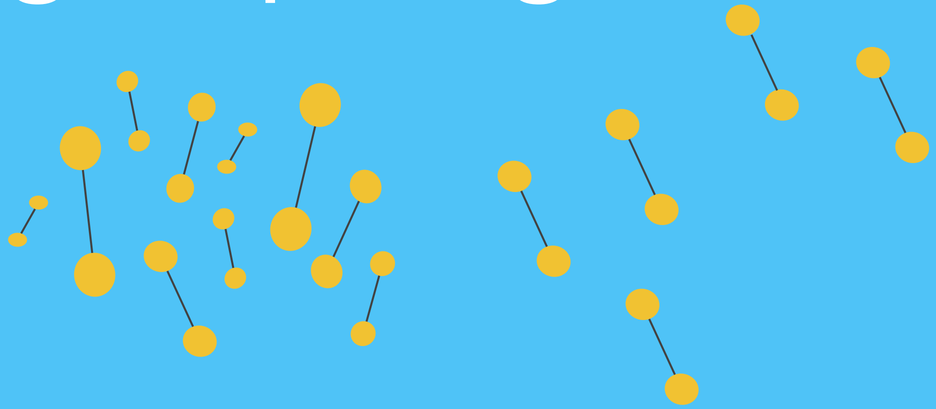
Revisiting NEAT: NeuroEvolution of Augmenting Topologies



Justin Chen

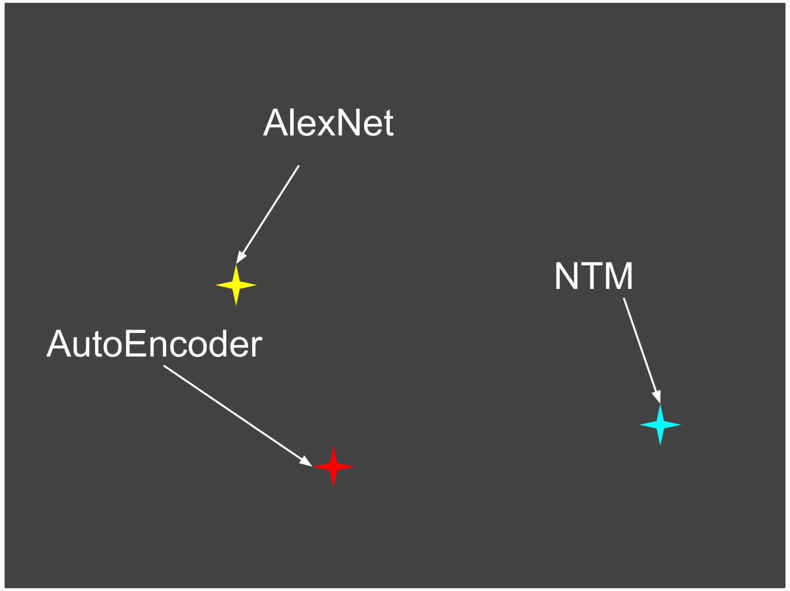
Automatic Model Selection

How is model selection currently done?

- Pruning
- Predefined architectures
- Domain knowledge required
- More or less guessing

What is automatic model selection?

- Automatic discovery of architecture for given task



Space of possible architectures

TWEANN:
Topology Weighted Evolution of ANN

- “Can evolving topologies along with weights provide an advantage over evolving weights on a fixed-topology?”
- NE **searches for a behavior** instead of optimizing cost function
- Effective in **continuous** and **high-dimensional** state space
- Uses **mutation**, **mating**, and **fitness score** for discovering topology and weights
- **Saves time** on deciding architecture

Objectives & Challenges

- Genetic representation that allows meaningful crossover?
- How to protect and optimize innovation?
- How to minimize topology without specially contrived fitness function?

NEAT

- *Kenneth Stanley and Risto Miikkulainen 2002*
- Minimize dimensionality of connection weight search space
- Key properties:
  1. **Minimal topology**
  2. **Speciation**
  3. **Principled crossover**

Encoding Schemes

Topology encoded in **genome**, which produce **phenotype** (physical topology)

Direct encoding

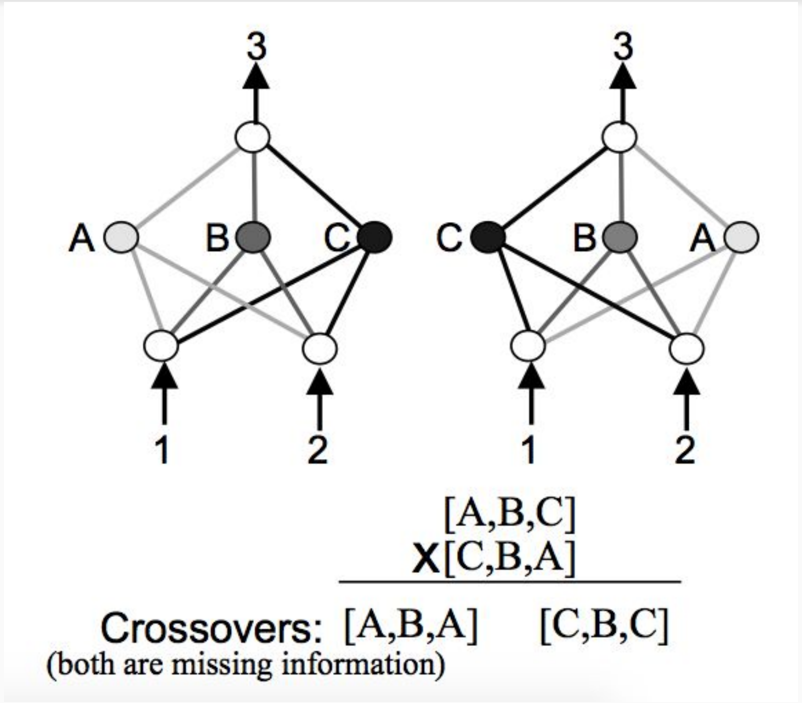
- **Explicit specification** of every connection and node in network
    - This is what NEAT uses

Indirect encoding

- Instructions that allow network topology to be **derived**

Competing Conventions

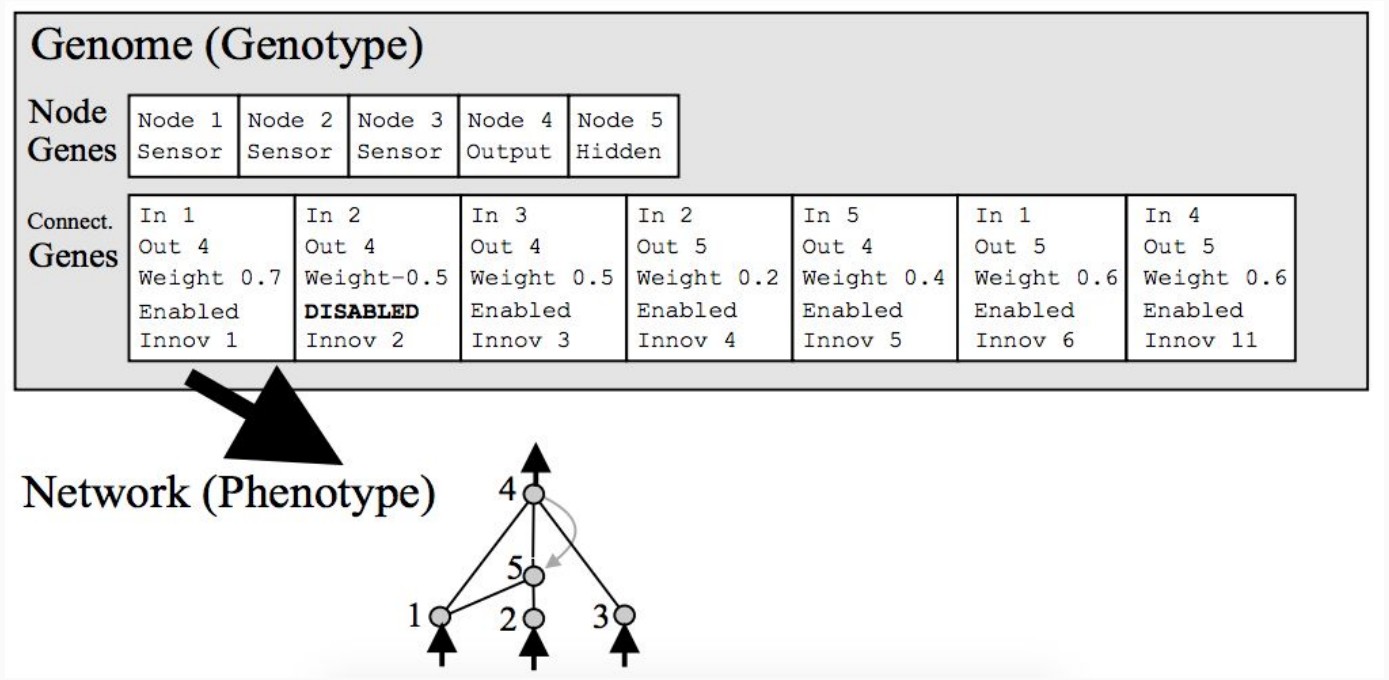
- aka Permutation Problem
- More than one way of expressing solution
- Lead to **damaged** (suboptimal) networks
- Can be caused by crossover, “mating” of neural networks



Historical Origin

- Direct evidence for homology
  - **Homology** - two representations are homologous if they share the **same origin**
- Allows adding new structure without losing track of where it came from
- Unique **innovation number** assigned to each gene
  - **Gene** - representation for node or connection
  - Number never changes throughout evolution
  - **Global variable** throughout evolution

Genetic Representation



Genome (Genotype)

Node Genes

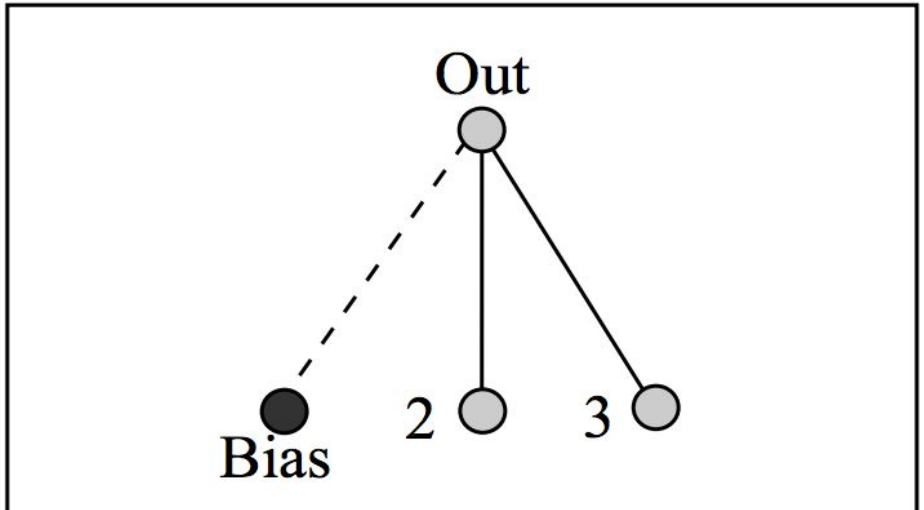
Node 1 Sensor	Node 2 Sensor	Node 3 Sensor	Node 4 Output	Node 5 Hidden

Connect. Genes

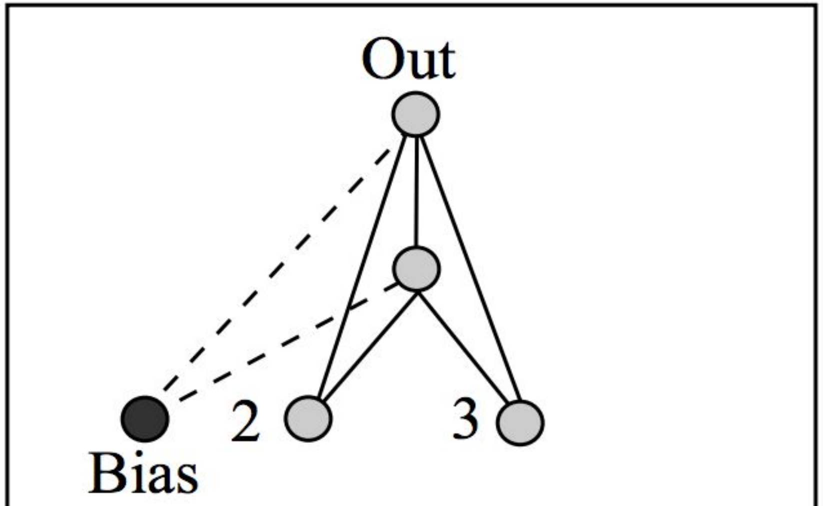
In 1	In 2	In 3	In 2	In 5	In 1	In 4
Out 4	Out 4	Out 4	Out 5	Out 4	Out 5	Out 5
Weight 0.7	Weight-0.5	Weight 0.5	Weight 0.2	Weight 0.4	Weight 0.6	Weight 0.6
Enabled	**DISABLED**	Enabled	Enabled	Enabled	Enabled	Enabled
Innov 1	Innov 2	Innov 3	Innov 4	Innov 5	Innov 6	Innov 11

Network (Phenotype)

Minimal Topology



Phenotype of all genomes in initial population
(No hidden nodes)



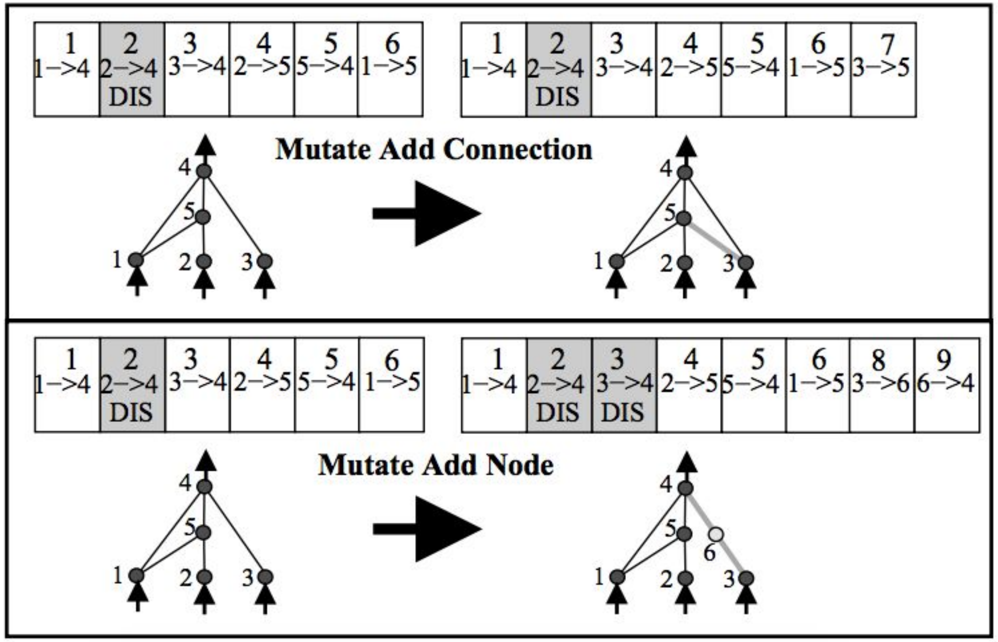
Phenotype of smallest possible solution

Minimal Topology

- Start with population with **simple topology**
  - e.g. One input node, one output node, no hidden nodes
  - Minimize architecture search space
- Complexify over evolution
  - **Justifies** each architectural component
- Possible to obtain **diversity** because NEAT employs **speciation**

Mutation

- New genes get **new innovation number**
- Add **connection**
  - Gene appended to genome
  - New connection with random weight
- Add **node**
  - Connection **split** and **disabled**
  - Append **two** connection genes
  - **In connection** gets **weight 1**
  - **Out connection unchanged**



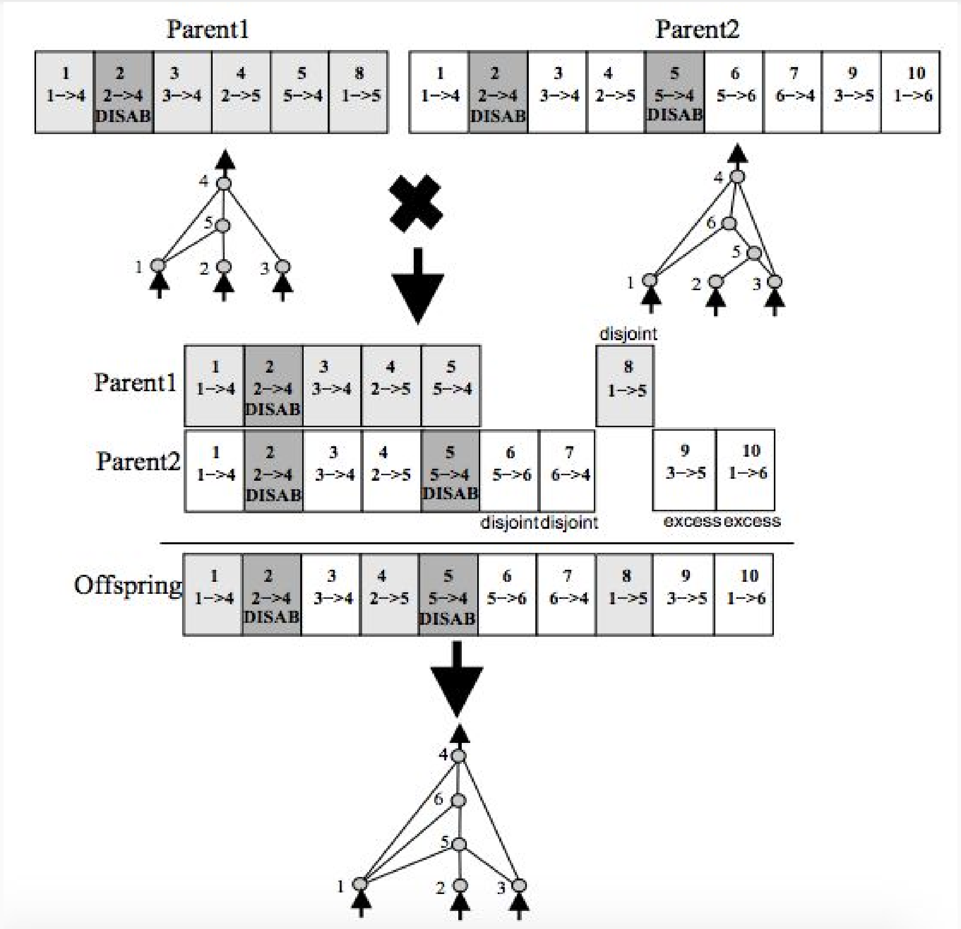
Tracking Historical Markings

- Genes with **similar historical markings** are more **topologically similar**
  - Can conclude from **same “family tree”**
- Requires little computation
- **Historical markings**:
  - Same mutations in same generation assigned **same innovation number**
  - Compare all previous innovation numbers in each individual
  - Controls growth of innovation numbers

Principled Crossover

- Sexual reproduction
- Two genomes **“crossover”** individual genes
- **Matching genes**
  - Using historical markings, same genes occurring in both genomes are aligned
- **Disjoint genes**
  - Genes with innovation numbers **less than** other genome’s highest innovation number
- **Excess genes**
  - Genes with innovation numbers **greater than** other genome’s highest innovation number
- Aligned genes inherited randomly from either parent
- Disjoint and Excess only inherited from more fit parent

Principled Crossover



Speciation

- **Compatibility function** - determine if individual (**genomes**) belongs in same niche
- **Explicit fitness sharing** - fitness shared by all in same niche based on genetic similarity
  - Measure using historical origins
- **Implicit fitness sharing** - fitness shared by all in niche based on behavior

Compatibility Distance

- Ratio of **shared genes** corresponds to **compatibility ratio**
- E = excess, D = disjoint, W = avg. weight difference of **all matching genes**
- N = number of genes in larger genome
- c_{i} = weight

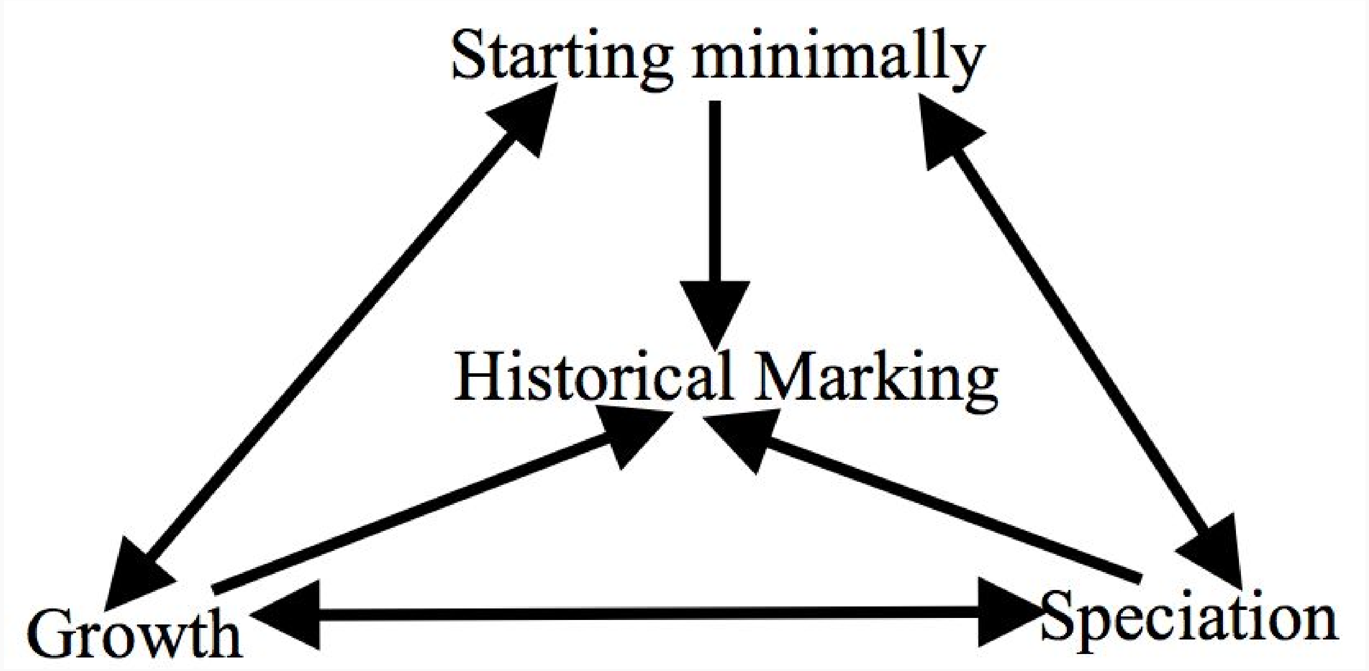
$$\delta = \frac{c_1 E}{N} + \frac{c_2 D}{N} + c_3 \cdot \overline{W}$$

Explicit Fitness Sharing

- All organisms in same species share a single fitness score
- Controls a single species from dominating all other species
- Adjusted fitness for organism ***i***

$$f'_i = \frac{f_i}{\sum_{j=1}^{n} \mathrm{sh}(\delta(i,j))}$$

Ablation Test

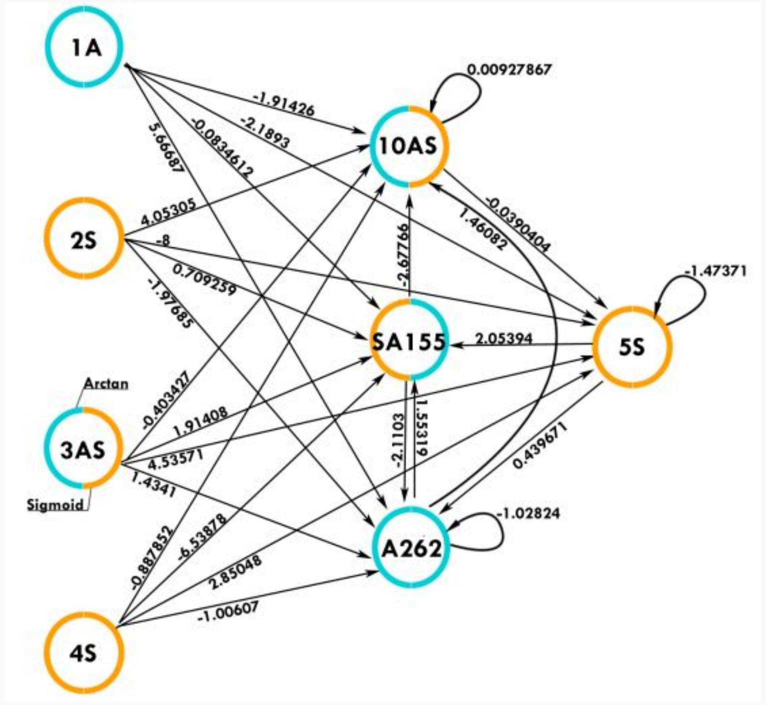


Advantages

1. No gradient optimization
2. Allows for more expressive models
3. Automatically discover appropriate architecture while simultaneously optimizing weights

Disadvantages

1. Architectures algorithm evolves are difficult to interpret, which makes it hard to figure out why something may have went wrong
2. Hyperparameter tuning:
   - Sensitive to initial population size
   - Compatibility distance coefficients
   - etc.



Large-Scale Evolution of Image Classifiers

LSE

- *Esteban Real, Sherry Moore, Andrew Selle, Saurabh Saxena, Yutaka Leon Suematsu, Quoc Le, Alex Kurakin, Google Brain, 2017*
- Key properties:
  1. **Simplicity**
  2. **Fully trained networks without post-processing**
  3. **Scalability**
- Train on CIFAR-10
- Try to generalize to CIFAR-100

Evolutionary Algorithm

- Start with simple networks - no convolution, and learning rate of 0.1
    - Linear regression models
- **Fitness** measure with **validation set**
- Tournament Selection
    - Each generation, two networks randomly selected and fitness scores compared
        - More fit network allowed to reproduce
        - Lesser network is **“killed”**
    - Child evaluated on validation set and returns to population

Scalability

- Massively-parallel, lock-free distributed system
- Store population on a shared file-system
    - Directories represent individuals
    - Renaming directories = change in state (alive or dead)
- Population size: 1000
- Workers: always *population size/4*
- Frequent garbage collection

Encoding and Mutation

- DNA of each individual represented as a graph
- **Vertices** = rank-3 tensor (activations)
  - Dimensions: height x width x channels
  - Optionally apply: batch norm with ReLU
- **Edges** = Convolution parameters or identity (no change to parameters)
- Mutation selected randomly from user-defined set
  - *Refer to page 4 for list of mutations*
- **No limit on possible mutations** (depth, convolutional parameters, etc.)
- Mutation components define search space

Training

- SGD
- momentum = 0.9
- batch = 50
- weight decay = 0.0001
- 25,600 steps
- Validation score used as individual’s fitness

Computation Cost

- Chose **basic TensorFlow operations** during training and validation
  - Convolution, matrix multiplication, etc.
- Measured operations in **FLOPs** (floating-point operations)
- Measure FLOPs in **training and validation** for each individual
- Assign cost of $F_t N_t E_t + F_v N_v$ to each network
  - F = FLOPs
  - E = number of training epochs
  - N = number of training and validation examples
  - t = training
  - v = validation

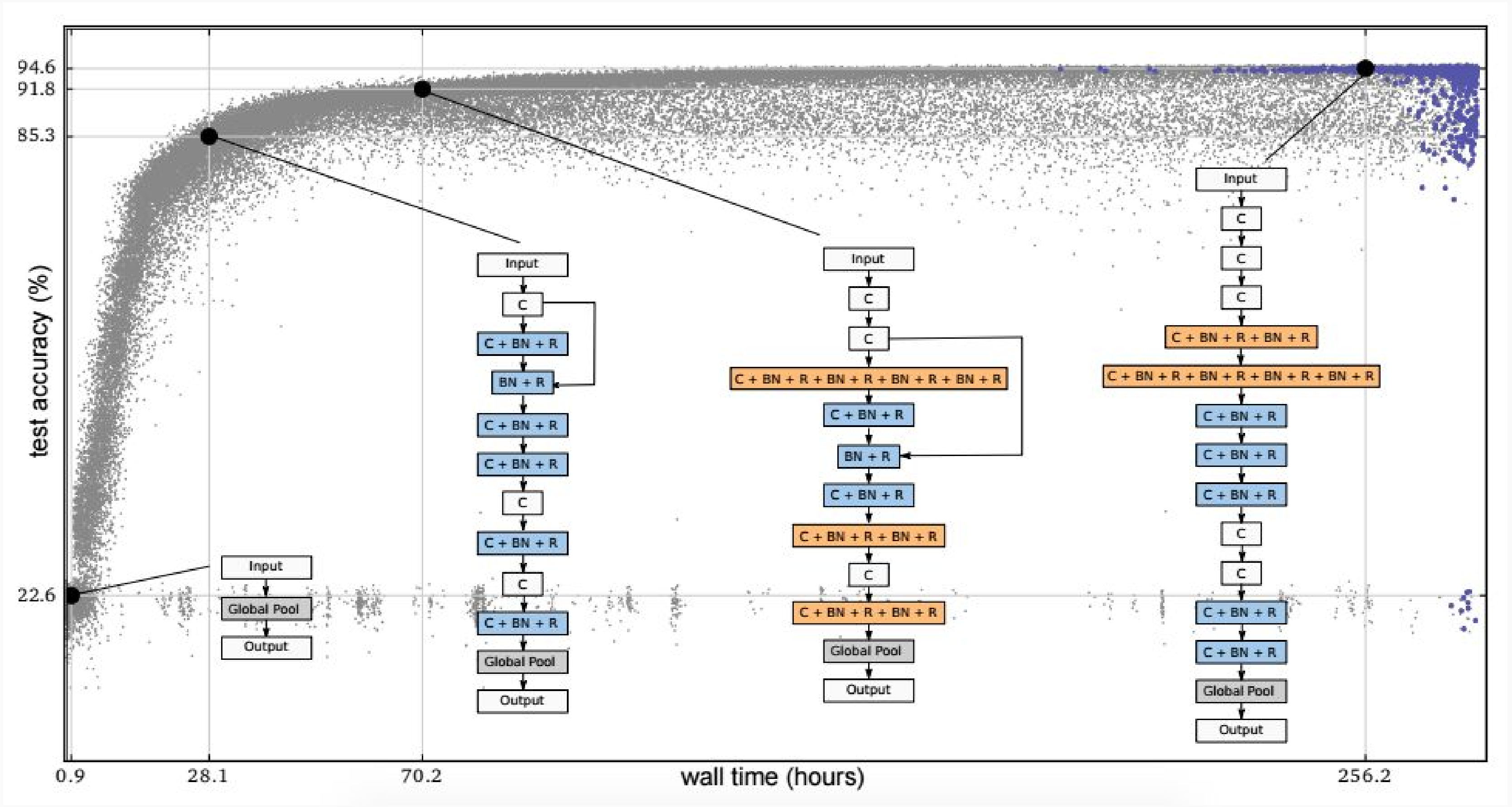
Weight Inheritance

- **Possible issue**
  - Architectures trained to completion in each evolutionary experiment
  - If not complete, best model is retrained
- **Solution**
  - Children inherit parents’ weights if layers are unchanged or mutated shapes match

Comparison to NEAT

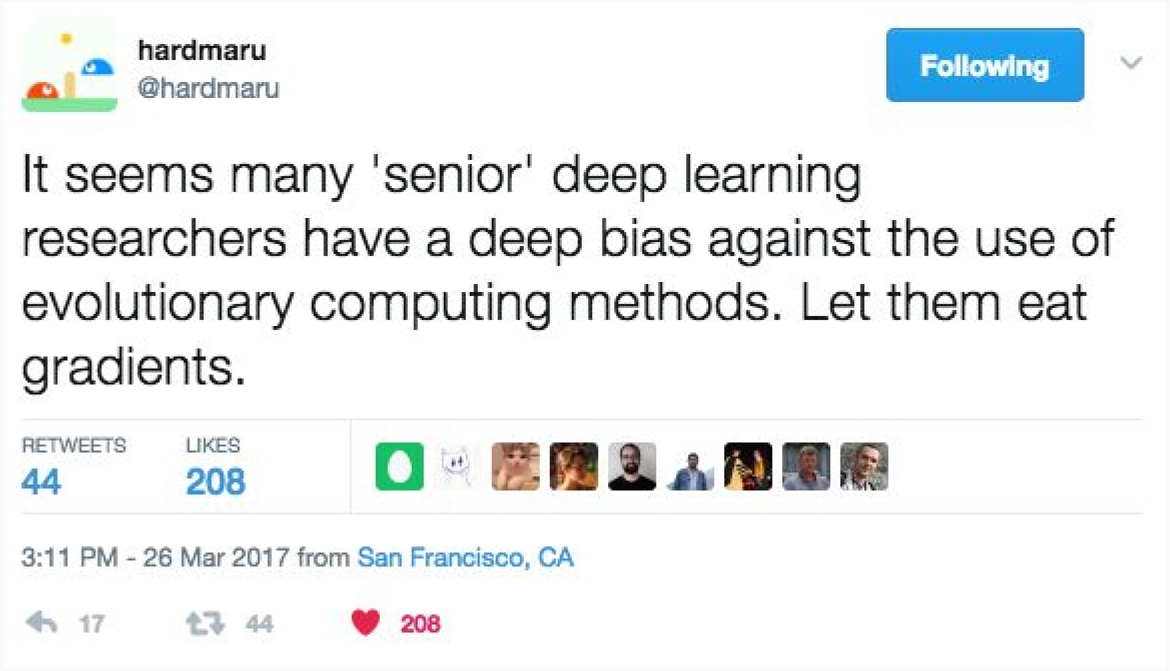
LSE	NEAT
- Start from minimal topology	- Start from minimal topology
- Mutate layers	- Mutate nodes
- Prune layers	- No pruning
- No fitness sharing	- Fitness sharing
- No speciation	- Speciation
- No mating (asexual)	- Mating/ Crossover (sexual)
- Backpropagation	- Random perturbations
- Learning rate mutation	

Progress of an evolution experiment



Evolution Strategies as a Scalable Alternative to Reinforcement Learning

Evolution v. Gradient Optimization



hardmaru
@hardmaru

Following

It seems many 'senior' deep learning researchers have a deep bias against the use of evolutionary computing methods. Let them eat gradients.

RETWEETS 44 LIKES 208

3:11 PM - 26 Mar 2017 from San Francisco, CA

17 44 208

Evolution Strategies

- Key properties:
  1. **Derivative-free hill-climbing**
  2. **Highly parallelizable**
  3. **Data-time trade-off**
  4. **Novel behavior discovery**
  5. **Fixed hyperparameters for all challenges in environment**
  6. **Indifference to types of reward signals and time horizons**

Evolution Strategies

- Goal: maximize average objective value $\eta(\psi) = E_{\theta \sim p_\psi} F(\theta)$
  - By searching for ψ using stochastic gradient ascent
- Initialize population
- Mutate parameters
- Evaluate
- Stochastic gradient estimate
- Parameter update

Mutation

- Generate offspring population by adding Gaussian noise to parameters
- Evaluate each mutated network in the environment
- Collect rewards for each network
- New parameter vector is weighted sum of offspring
    - Better performing networks are weighed more

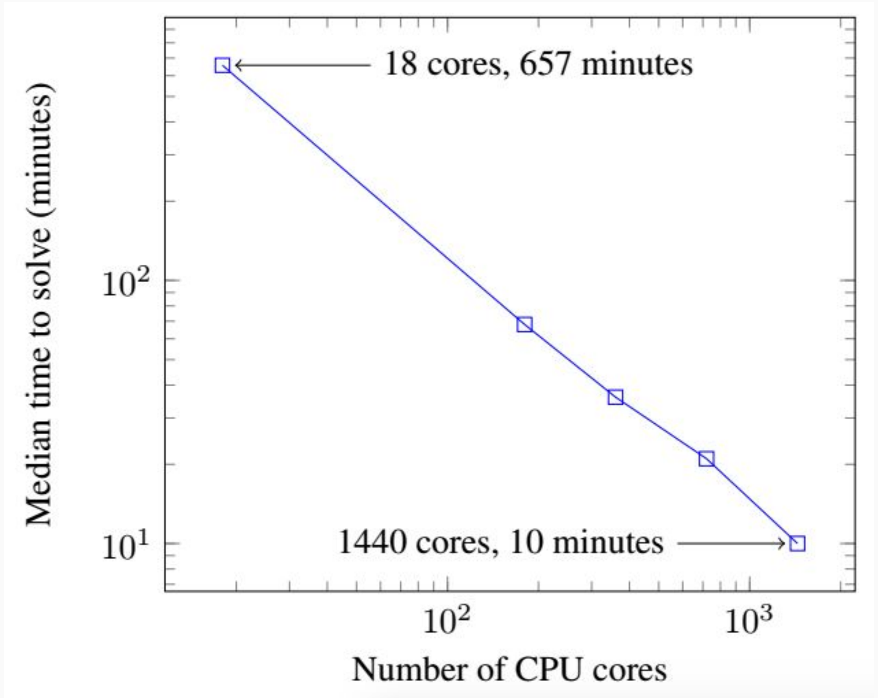
Natural Evolution Strategy

- *Daan Wierstra, Tom Schaul, Jan Peters, Juergen Schmidhuber, 2008*
- Individual represented as a n-dimensional vector
- Initialize each individual with multivariate norm
  - Places individuals randomly on fitness landscape
  - λ starting points
  - Randomly mutate and choose best μ
  - Mutated are evaluated = fitness score on given task

Performance

- Cluster with 80 machines 1440 CPUs
- Outperformed A3C on 23 Atari games
- Underperformed A3C on 28 games
- 3x to 10x more data for Atari
- 10x more data for MuJoCo than Trust Region Policy Optimization
- 1 hour to reach same accuracy on Atari, which took A3C1 day

Linear scalability



ES v. RL

Evolution Strategies

1. Search parameter space
2. Non-differentiable operations
3. Communicate scalars (reward)
4. Robust to frame-skipping
5. Better exploration
6. Better credit assignment
7. Needs more data
8. Faster

Reinforcement Learning

1. Search action space
2. Backpropagation
3. Communicate entire gradients
4. Reward signal-depends on frame rate
5. Tends towards greedy behavior

Citations

[1] Stanley, Kenneth O., and Risto Miikkulainen. "Evolving neural networks through augmenting topologies." Evolutionary computation 10.2 (2002): 99-127.

[2] Real, Esteban, et al. "Large-Scale Evolution of Image Classifiers." arXiv preprint arXiv:1703.01041 (2017).

[3] Salimans, Tim, et al. "Evolution Strategies as a Scalable Alternative to Reinforcement Learning." arXiv preprint arXiv:1703.03864 (2017).

[4] Wierstra, Daan, et al. "Natural evolution strategies." Evolutionary Computation, 2008. CEC 2008.(IEEE World Congress on Computational Intelligence). IEEE Congress on. IEEE, 2008.