Evolution of Neural Networks

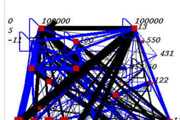
Risto Miikkulainen

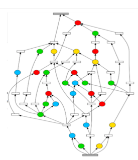
The University of Texas at Austin
and Cognizant Technology Solutions

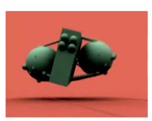
GECCO '19 Companion, July 13–17, 2019, Prague, Czech Republic

ACM ISBN 978-1-4503-6748-6/19/07.
https://doi.org/10.1145/3319619.3323380

Why Evolve Neural Networks?





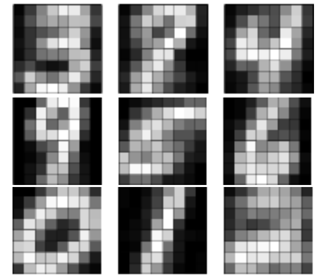


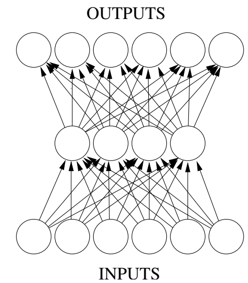
I Original role (since 1990s): Sequential Decision Tasks
- Both the structure and the weights evolved (no training)
- Power from recurrency: POMDP tasks; behavior

II A new role (since 2016): Optimization of Deep Learning Nets
- Components, topology, hyperparameters evolved; weights trained
- Power from complexity

III A possible future role: Emergence of intelligence
- Body/brain co-evolution; Competitive co-evolution
- Evolution of memory, language, learning

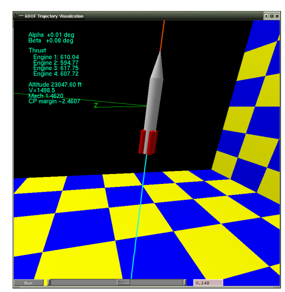
Why Use Neural Networks?





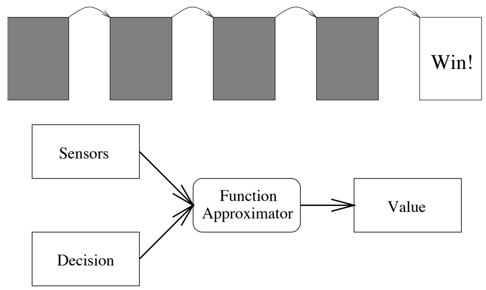
- Neural nets powerful in many statistical domains
  - E.g. control, pattern recognition, prediction, decision making
  - Where no good theory of the domain exists
- Good supervised training algorithms exist
  - Learn a nonlinear function that matches the examples
  - Utilize big datasets

I. Sequential Decision Tasks



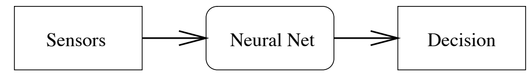
- A sequence of decisions creates a sequence of states
  - States are only partially known
  - Optimal outputs are not known
  - We can only tell how well we are doing
- Exist in many important real-world domains
  - Robot/vehicle/traffic control
  - Computer/manufacturing/process optimization
  - Game playing; Artificial Life; Biological Behavior

Standard Reinforcement Learning



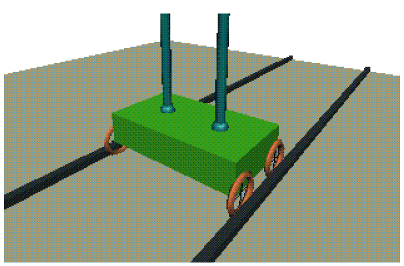
- AHC, Q-learning, Temporal Differences
  - Generate targets through prediction errors
  - Learn when successive predictions differ
- Predictions represented as a value function
  - Values of alternatives at each state
- Difficult with large/continuous state and action spaces
- Difficult with hidden states

Neuroevolution (NE) Reinforcement Learning



- NE = constructing neural networks with evolutionary algorithms
- Direct nonlinear mapping from sensors to actions
- Large/continuous states and actions easy
  - Generalization in neural networks
- Hidden states (in POMDP) disambiguated through memory
  - Recurrency in neural networks[79]
  - Deep Reinforcement Learning[54,63]

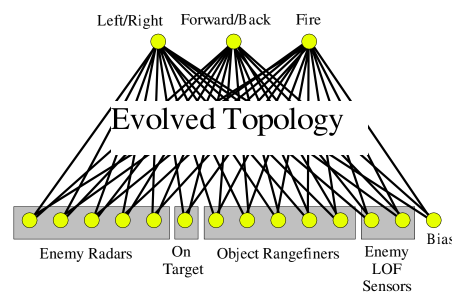
How Well Does It Work?



Poles	Method	Evals	Succ.
One	VAPS	(500,000)	0%
	SARSA	13,562	59%
	Q-MLP	11,331	
	NE	127	
Two	NE	3,416	

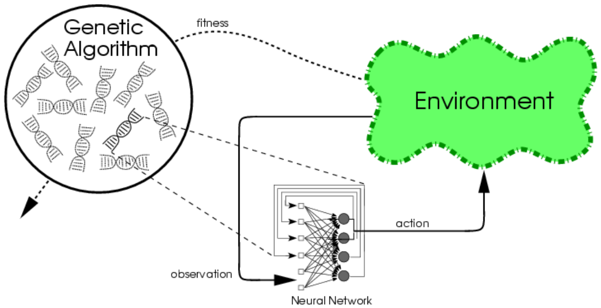
- Difficult RL benchmark: POMDP Pole Balancing
- NE 2-3 orders of magnitude faster than standard RL[21]
- NE can solve harder problems

Neuroevolution for POMDP



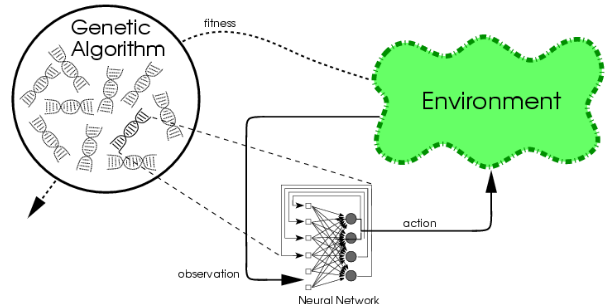
- Input variables describe the state observed through sensors
- Output variables describe actions
- Network between input and output:
  - Recurrent connections implement memory
  - Memory helps with POMDP

Basic Neuroevolution (1)



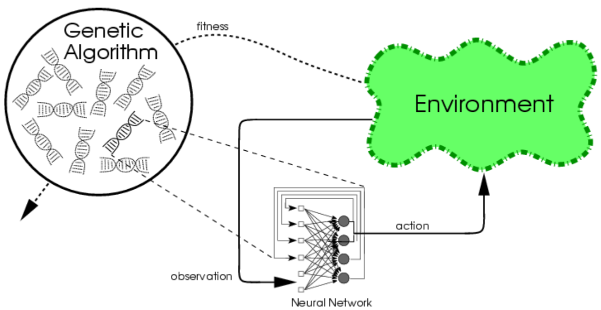
- Evolving connection weights in a population of networks [47,62,89,90]
- Chromosomes are strings of connection weights (bits or real)
  - E.g. 10010110101100101111001
  - Usually fully connected, fixed topology
  - Initially random

Basic Neuroevolution (2)



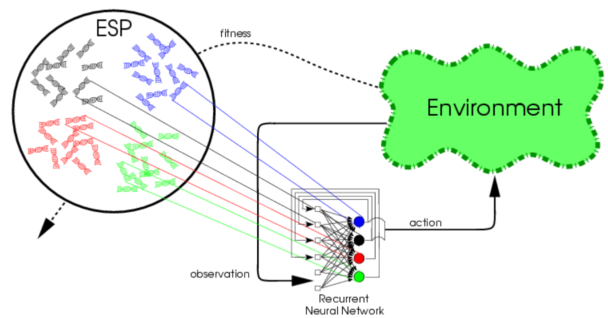
- Parallel search for a solution network
  - Each NN evaluated in the task
  - Good NN reproduce through crossover, mutation
  - Bad thrown away
- Natural mapping between genotype and phenotype
  - GA and NN are a good match!

Problems with Basic Neuroevolution



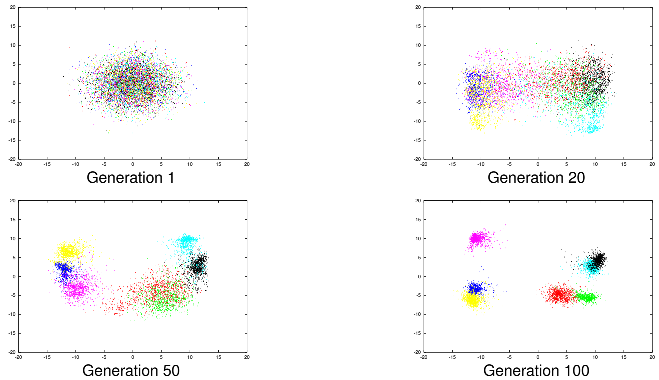
- Evolution converges the population (as usual with EAs)
  - Diversity is lost; progress stagnates
- Competing conventions
  - Different, incompatible encodings for the same solution
- Too many parameters to be optimized simultaneously
  - Thousands of weight values at once

Advanced NE 1: Evolving Partial Networks



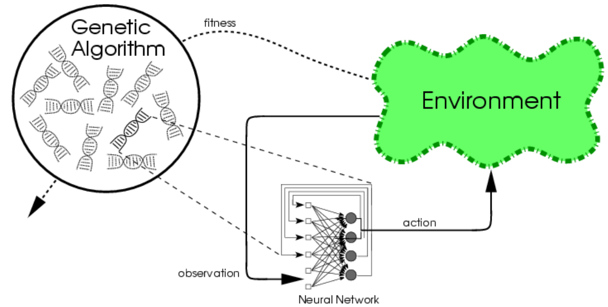
- Evolving individual neurons to cooperate in networks [1,48,53]
- E.g. Enforced Sub-Populations (ESP [18])
  - Each (hidden) neuron in a separate subpopulation
  - Fully connected; weights of each neuron evolved
  - Populations learn compatible subtasks
- Can be applied at the level of weights, and modules

Evolving Neurons with ESP



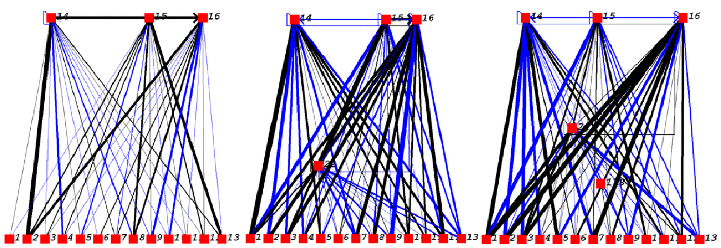
- Evolution encourages diversity automatically
  - Good networks require different kinds of neurons
- Evolution discourages competing conventions
  - Neurons optimized for compatible roles
- Large search space divided into subtasks
  - Optimize compatible neurons

Advanced NE 2: Evolutionary Strategies



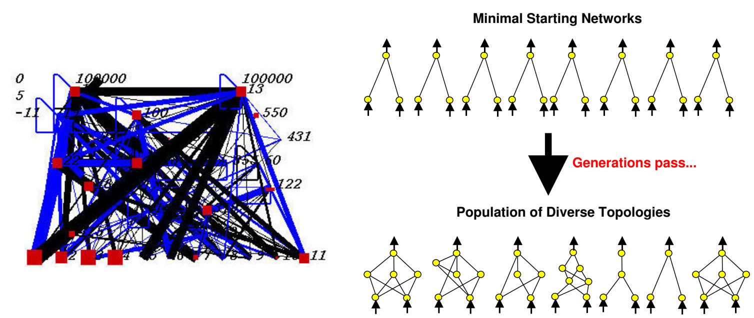
- Evolving complete networks with ES (CMA-ES[26])
- Small populations, no crossover
- Instead, intelligent mutations
  - Adapt covariance matrix of mutation distribution
  - Take into account correlations between weights
- Smaller space, less convergence, fewer conventions

Advanced NE 3: Evolving Network Structure



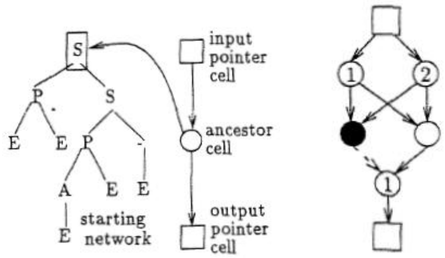
- Optimizing connection weights and network topology[2,14,16,92]
- E.g. Neuroevolution of Augmenting Topologies (NEAT[70,73])
- Based on *Complexification*
- Of networks:
  - Mutations to add nodes and connections
- Of behavior:
  - Elaborates on earlier behaviors

Why Complexification?



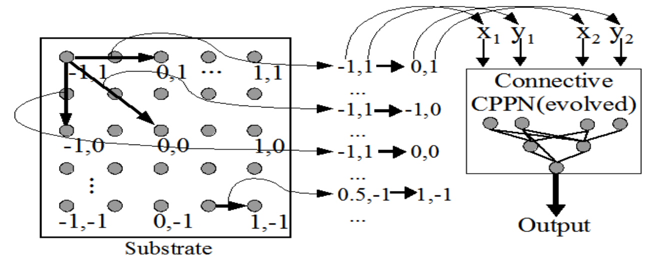
- Challenge with NE: Search space is very large
- Complexification keeps the search tractable
  - Start simple, add more sophistication
- Incremental construction of intelligent agents

Advanced NE 4: Indirect Encodings (1)

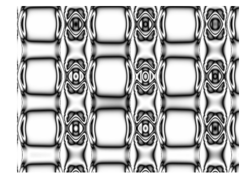


- Instructions for constructing the network evolved
  - Instead of specifying each unit and connection[2,14,46,68,92]
- E.g. Cellular Encoding (CE[23])
- Grammar tree describes construction
  - Sequential and parallel cell division
  - Changing thresholds, weights
  - A “developmental” process that results in a network

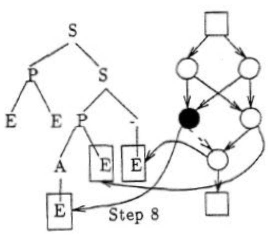
Indirect Encodings (2)

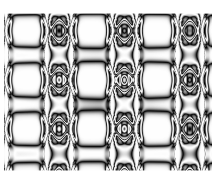


- Encode the networks as spatial patterns
- E.g. Hypercube-based NEAT (HyperNEAT[8])
- Evolve a neural network (CPPN) to generate spatial patterns
  - 2D CPPN: (x, y) input → grayscale output
  - 4D CPPN: (x_1, y_1, x_2, y_2) input → w output
  - Connectivity and weights can be evolved indirectly
  - Works with very large networks (millions of connections)



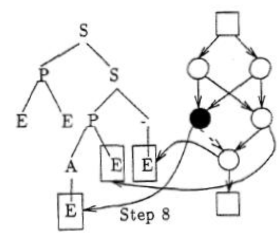
Properties of Indirect Encodings (1)

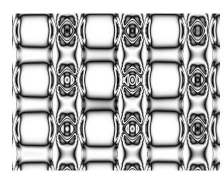




- Smaller search space
- Avoids competing conventions
- Describes classes of networks efficiently
- Modularity, reuse of structures
  - Recurrency symbol in CE: XOR → parity
  - Repetition with variation in CPPNs
  - Useful for evolving morphology

Properties of Indirect Encodings (2)



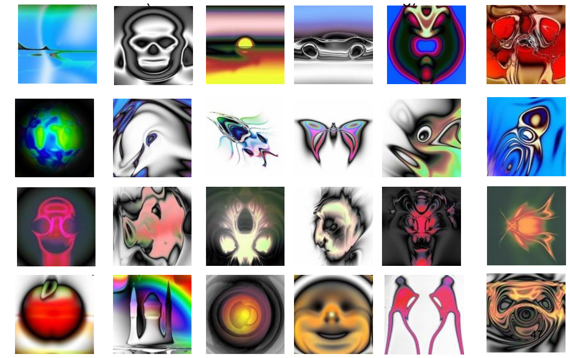


- Not fully explored (yet)
  - See e.g. CS track at GECCO
- Promising current work
  - More general L-systems; developmental codings; embryogeny[74]
  - Scaling up spatial coding[9,17]
  - Genetic Regulatory Networks[58]
  - Evolution of symmetries[82]

Further NE Techniques

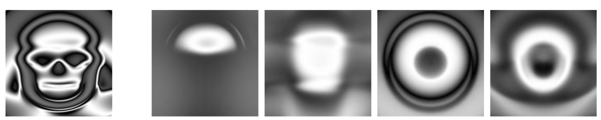
- Incremental and multiobjective evolution[20,65,81,90]
- Utilizing population culture[4,42,78]
- Utilizing evaluation history[39]
- Evolving NN ensembles and modules[27,38,52,59,86]
- Evolving transfer functions and learning rules[6,60,75]
- Bilevel optimization of NE[37]
- Evolving LSTMs for strategic behavior[34]
- Combining learning and evolution[5,15,42,51,71,78,87]
- Evolving for novelty

Evolving for Novelty



- Motivated by humans as fitness functions
- E.g. picbreeder.com, endlessforms.com[66]
  - CPPNs evolved; Human users select parents
- No specific goal
  - Interesting solutions preferred
  - Similar to biological evolution?

Novelty Search

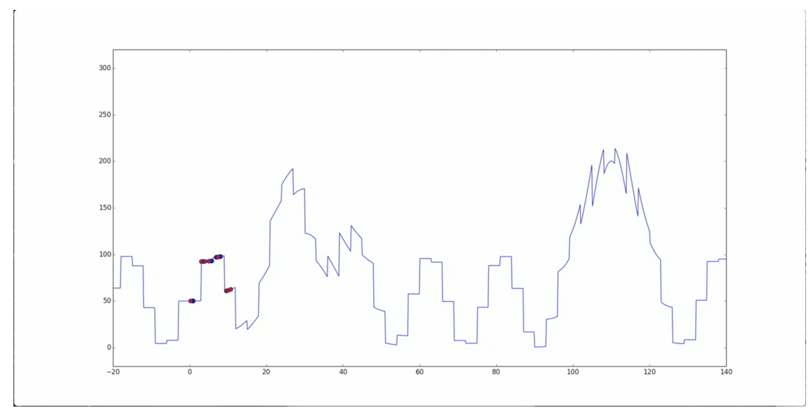


23f, 57c 74 gen | 20f, 24c failed | 20f, 29c failed | 19f, 24c failed | 22f, 28c failed

gen 12 | gen 20 | gen 36 | gen 49 | gen 74

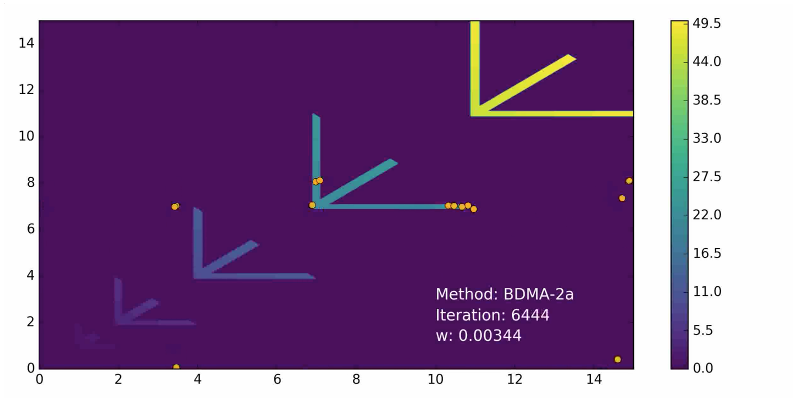
- Evolutionary algorithms maximize a performance objective
  - But sometimes hard to achieve it step-by-step
- Novelty search rewards candidates that are simply different[31,72]
  - Stepping stones for constructing complexity

Novelty Search Demo (1)



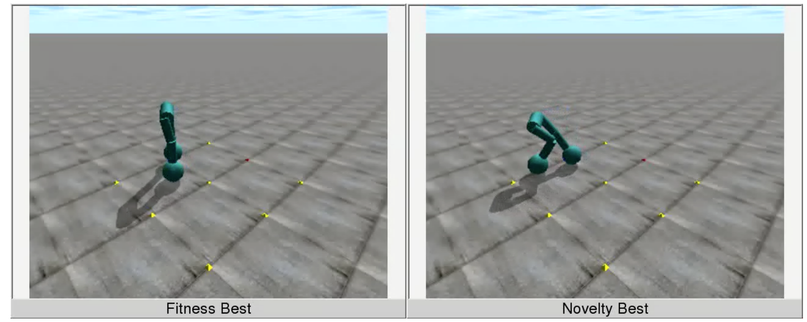
- 1D function to optimize; Fitness-based search would converge
- Novelty search finds stepping stones
- DEMO

Novelty Search Demo (2)



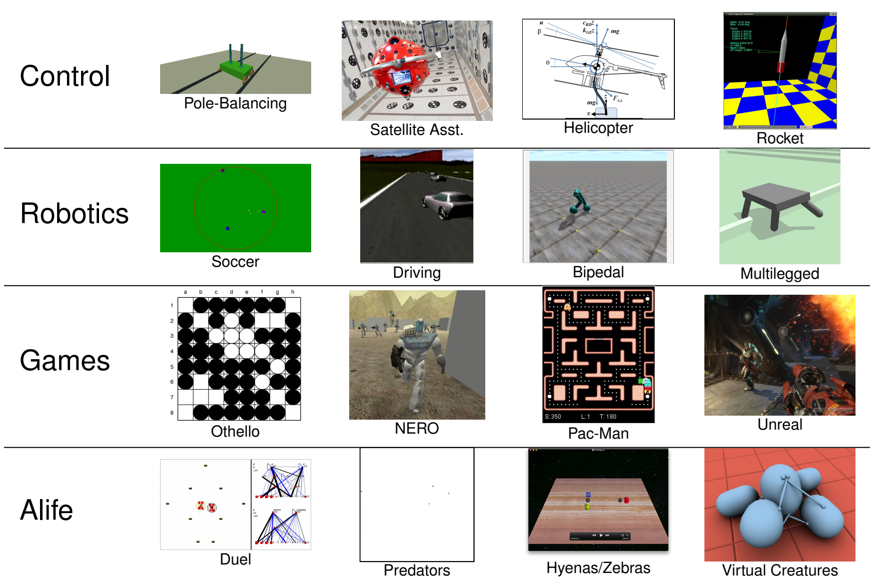
- Illustration of stepping stones[43,44]
  - Nonzero fitness on "feet" only; stepwise increase
  - Top and right "toes" are stepping stones to next "foot"
  - Difficult for fitness based search; novelty can do it
- DEMO

Novelty Search Demo (3)

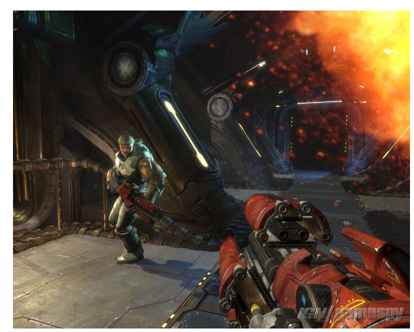


- Fitness-based evolution is rigid
  - Requires gradual progress
- Novelty-based evolution is more innovative, natural[31,72]
  - Allows building on stepping stones
  - As a secondary objective—or even the only one!
- DEMO

Neuroevolution Applications



Games: Evolving Humanlike Behavior

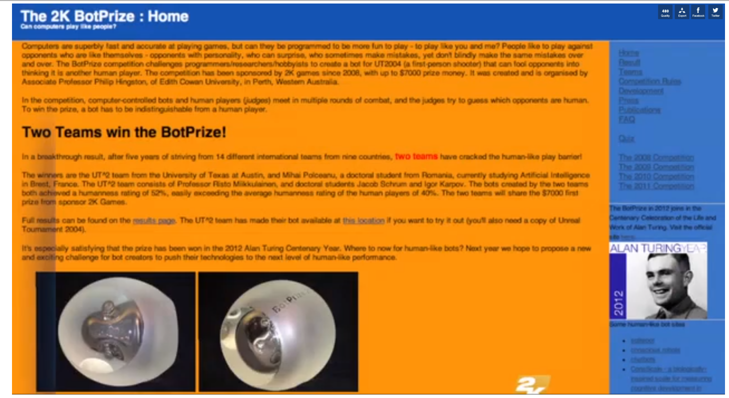


- Botprize competition, 2007-2012
  - Turing Test for game bots ($10,000 prize)
- Three players in Unreal Tournament 2004:
  - Human confederate: tries to win
  - Software bot: pretends to be human
  - Human judge: tries to tell them apart!

Evolving an Unreal Bot

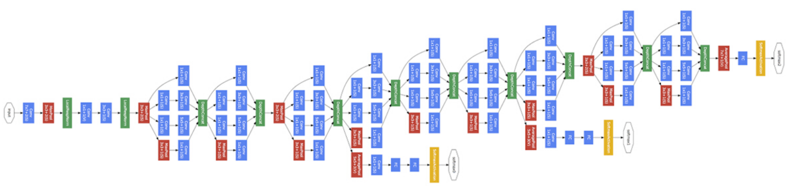
- Evolve effective fighting behavior
  - Human-like with resource limitations (speed, accuracy...)
- Also scripts & learning from humans (unstuck, wandering...)
- 2007-2011: bots 25-30% vs. humans 35-80% human
- 6/2012 best bot better than 50% of the humans
- 9/2012...?

Success!!!



- In 2012, two teams reach the 50% mark!
- Fascinating challenges remain:
  - Judges can still differentiate in seconds
  - Judges lay cognitive, high-level traps
  - Team competition: collaboration as well
- DEMO

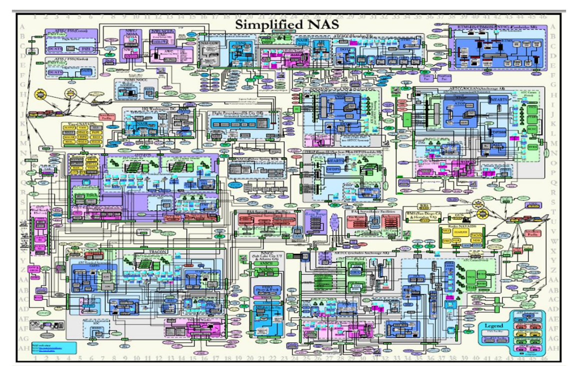
II. Optimization of DL Architectures



Szegedy et al. 2014

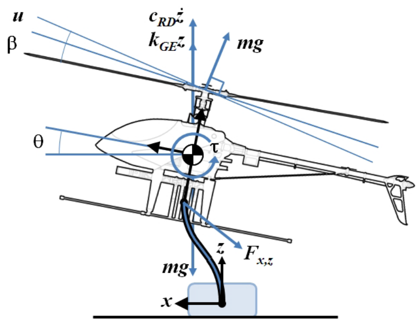
- Big Data and Big Compute available since 2000s
  - Machine learning systems have scaled up
- E.g. Deep Learning ideas existed since the 1990s
  - With million times more data & compute, they now work!
- A new problem: How to configure such systems?

Configuring Complex Systems



- A new general approach to engineering
  - Humans design just the framework
  - Machines optimize the details
- Programming by optimization[24]

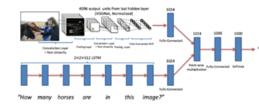
E.g. Optimizing NE in Helicopter Hovering



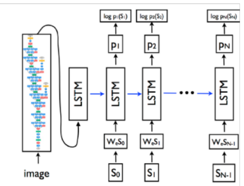
- A challenging benchmark
  - RL, NE solutions exist
- Eight parameters optimized by hand[29]
  - Hard for a human designer to do more
- With EA, increased to 15
  - →Significantly better performance[37]

Motivation for Neural Architecture Search

Szegedy et al. 2014



Agrawal et al. 2016

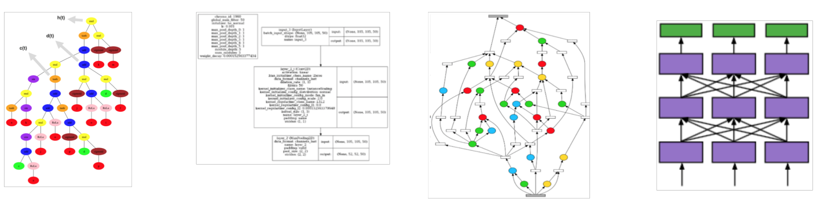


(Vinyals et al. 2016)

Different Deep Learning tasks require different architectures
- Architecture matters
- Too complex to optimize by hand

Employ neural architecture search (NAS)
Offer as a service in cloud computing?

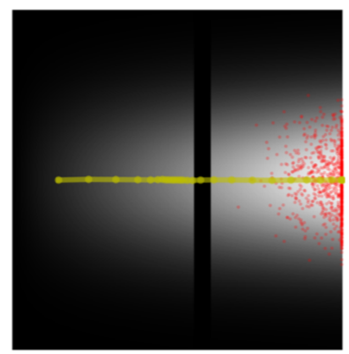
Motivation for Evolutionary NAS



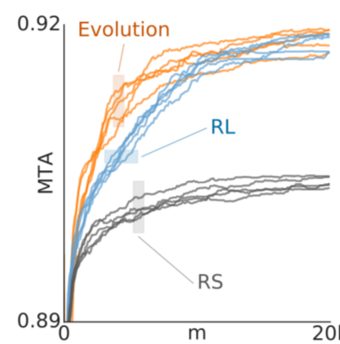
Evolutionary Neural Architecture Search is a natural fit:
- Population-based search covers the space
- Crossover between structures discovers principles
- Novelty search maximizes exploration

Building on Neuroevolution work since the 1990s
Hyperparameters; nodes; modules; topologies; multiple tasks

State of the Art Results in 2018



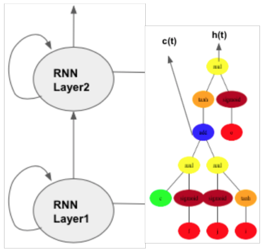
Lehman et al. 2017



Real et al. 2018

- Understanding ES and GAs in RL (Uber)[30,76,93]
  - ES provides more exploration than gradients
  - GA provides more exploration than ES
- Image processing (Google Brain)
  - CIFAR-10, CIFAR-100, and ImageNet[56]
- Language modeling and multitasking (Sentient)

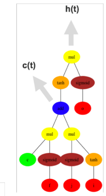
Node-level Evolution: Sequences



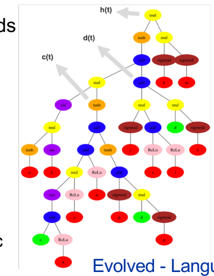
- Evolving gated memory units (i.e. LSTMs) for a fixed architecture
  - LSTM structure essentially the same for 25 years
- Tree representation of the nodes
  - Optimized through genetic programming

Node Evolution Benchmarks

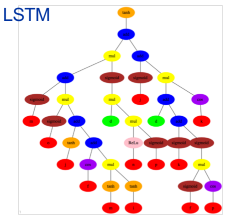
- Discovered multiple memory cells, nonlinear paths
  - Complexity matters!
  - Broader search than other NAS methods
- Language Modeling benchmark
  - Improve perplexity by 15%
  - State of the art in 1/2018
- Music Generation benchmark
  - A different solution necessary
  - Demo: ai.cognizant.com/evoai/lstm-music



LSTM



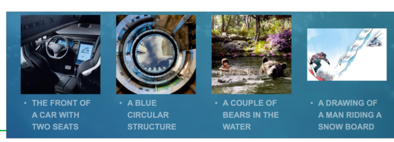
Evolved - Language

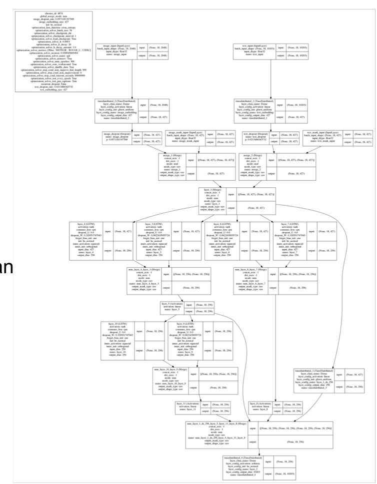


Evolved - Music

Improve Human Design

- E.g. image captioning:
  - Start with a state-of-the art design: Show&Tell
  - Search in the space of similar elements
  - 5% improvement
  - A prototype service on the web
- Best-performing AI defies human notions of symmetry an patterns of organization
- AI designing AI: could we automate it?

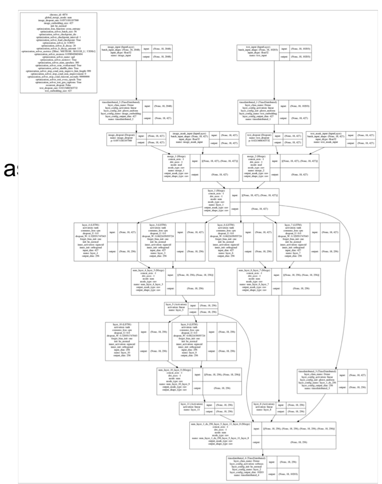




2019: Evolutionary AutoML

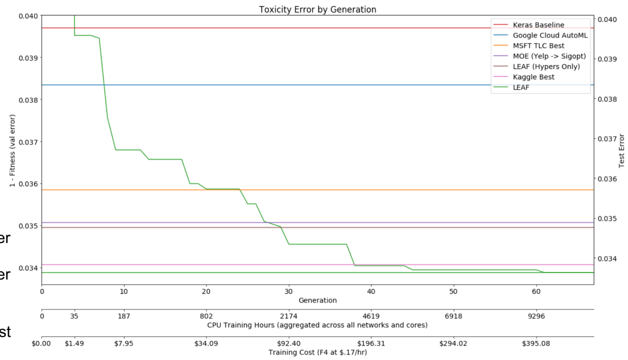
Current AutoML: Hyperparameter optimization
Evolutionary AutoML: Architectures and modules a

1. Improve over naïve baseline
   20% or more with little effort
2. Improve state of the art
   With more expertise & compute
3. Extend small datasets
   Multitasking with related datasets
4. Minimize network resources
   Train and run networks faster

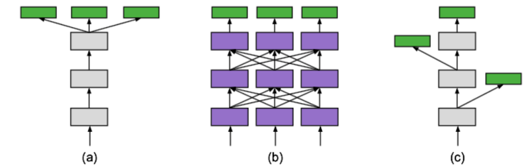


1 and 2: Improve Performance

- Domain: Wikipedia Toxic Comment Identification
  - Why: Toxicity is bad for business
  - Data: 160K labeled comments
  - Challenge: highly diverse vocabulary, style, and length
- Layer Types: Conv1D, LSTM, GRU
- LEAF Results:
  - With minimal compute: Improves over naïve Keras baseline
  - With more compute: Improves over other AutoML methods
  - With more compute: Improves over SOTA hand-designed model.
  - LEAF Hyperparameter Search on final architecture gives a final boost

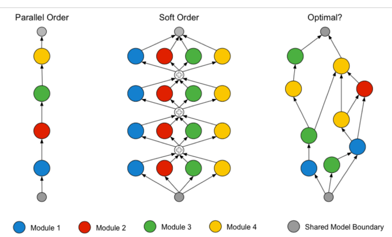


Network-level Evolution: Multitasking



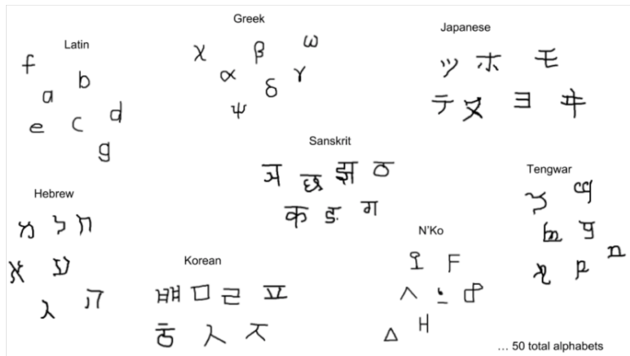
- Learning in multiple tasks at once
  - More generalizable embeddings
  - Each task can learn better
- Network structure can have a large effect
  - A good domain to test neuroevolution of structure

Network = Topology and Modules



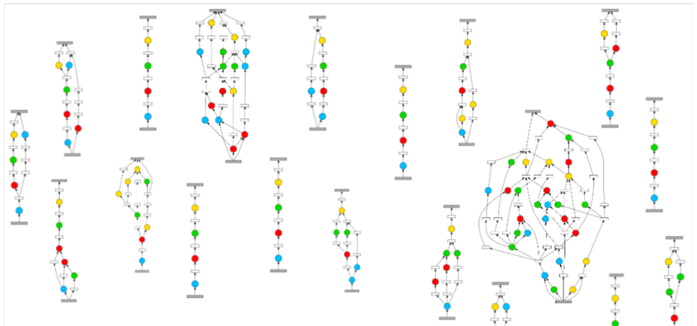
- Co-evolve the topology and modules
  (Liang et al. 2018; Meyerson et al. 2018)
- Different topology for each task, using same modules
- Modules trained across all tasks

3. Extend Small Datasets



- Recognize characters in 50 alphabets

Evolution of Multitask Networks: Topology+Modules

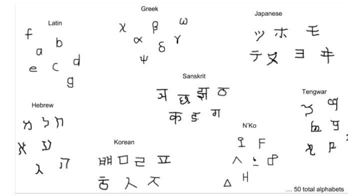


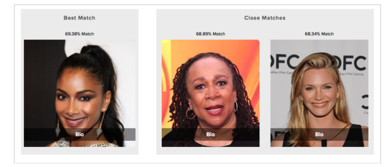
Multitasking Benchmarks

State-of-the-art in two ML benchmarks:

- Omniglot multialphabet character recognition
  - Improved state-of-the-art 31%
  - Demo: ai.cognizant.com/evoai/omni-draw
- CelebA multiattribute face classification
  - Improved state-of-the-art 0.75%
  - Demo: ai.cognizant.com/evoai/celeb-match

Improves learning in each task
- Even when little data available



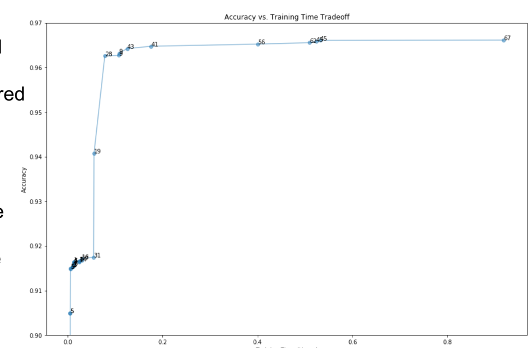


4. Minimize Network Resources

Evolution adds complexity only if needed
- Favors minimal solutions
- Over evolution a range of sizes explored
- Approximation of the Pareto front

Small networks found that perform well
- Minimization with little cost
- E.g. 0.38% drop with 1/12th of the size

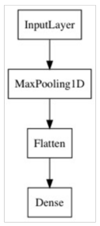
Adding a size objective will explore more



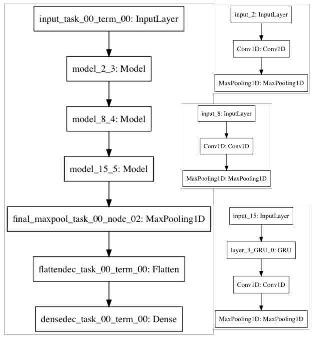
Multiobjective Minimization

- Animation: Pareto front by generation for single-objective (green) vs. multi-objective (blue)
- Single-objective focuses on improving *largest networks*
- Multi-objective focuses on improving the *entire curve*
- Result: Multi-objective finds much smaller models for the majority of performance values.

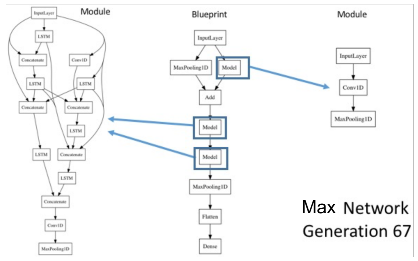
Example Performance/Size Tradeoffs



Min Network:
Generation 0

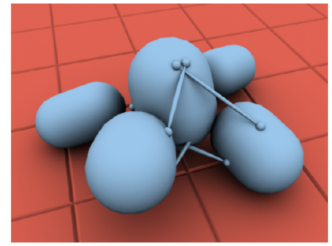


Tradeoff Network: Generation 28
- Sequential, GRUs after 2 pooling layers

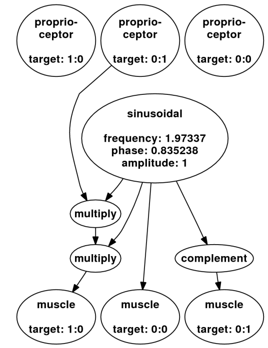


Max Network
Generation 67

III. Emergence of Intelligence



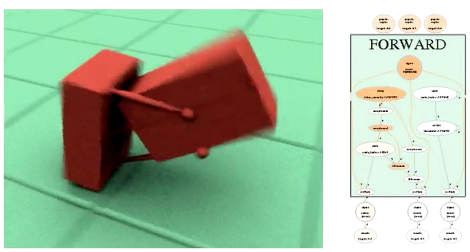
Body



Brain

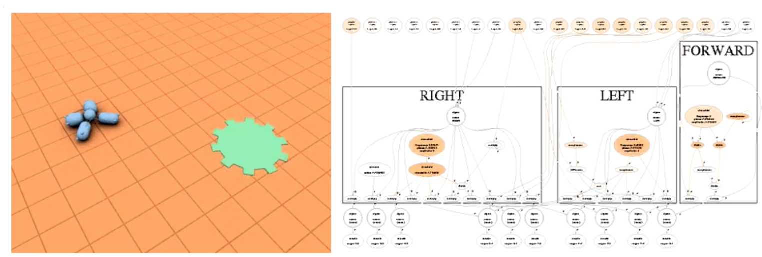
- Origins of intelligence: Embodied optimization
- Body-Brain Coevolution[1,2,3]
  - Body: Blocks, muscles, joints, sensors
  - Brain: A neural network (with general nodes)
  - Evolved together in a physical simulation
- Encapsulation, Pandemodium, Syllabus

Encapsulation



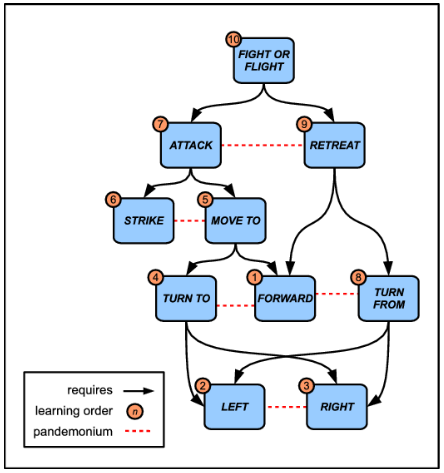
- Once evolved, a trigger node is added
- DEMO

Pandemonium



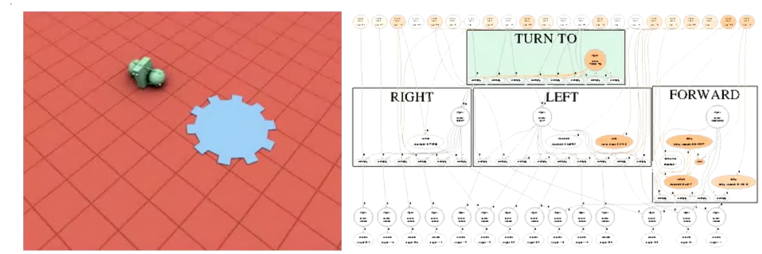
- Conflicting behaviors: Highest trigger wins
- DEMO

Syllabus



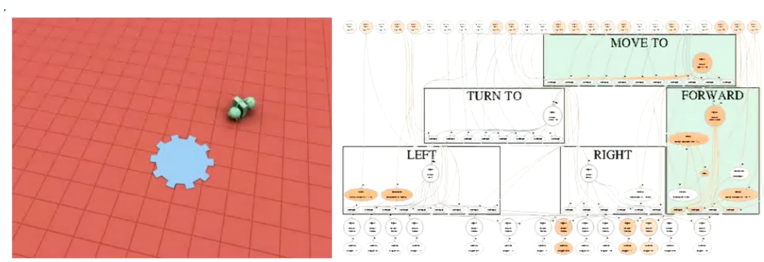
- Step-by-step construction of complex behavior
- Primitives and three levels of complexity
- Constructed by hand; body and brain evolved together

Turn to Light



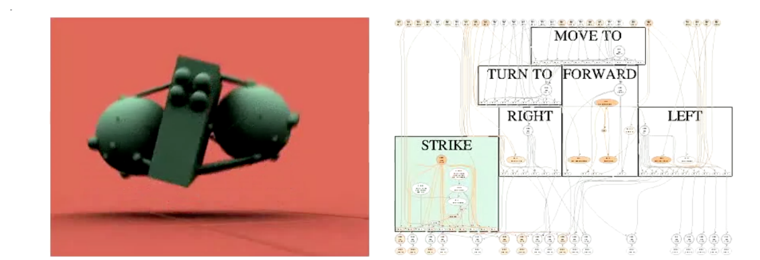
- First level of complexity
- Selecting between alternative primitives

Move to light



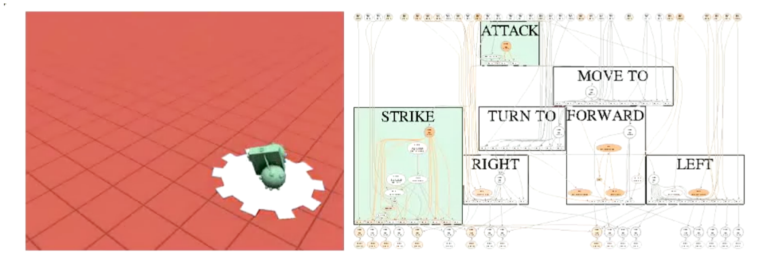
- First level of complexity (Sims 1994)
- Selecting between alternative primitives

Strike



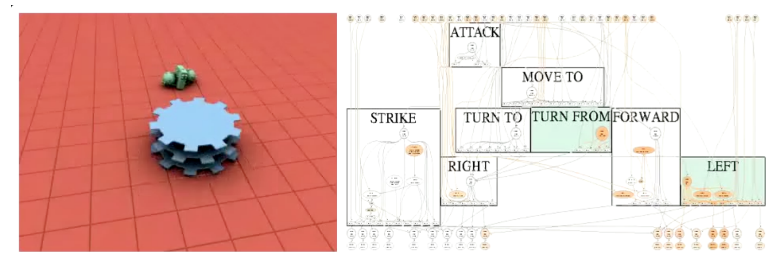
- Alternative behavior primitive

Attack



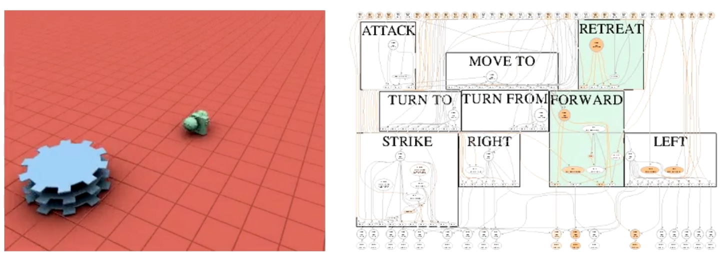
- Second level of complexity (beyond Sims and others)

Turn from Light



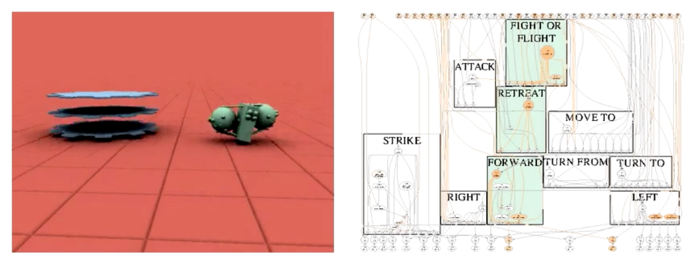
- Alternative first-level behavior

Retreat



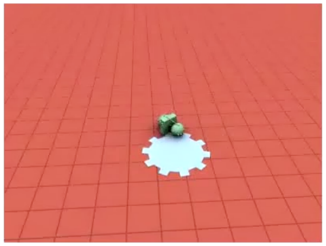
- Alternative second-level behavior

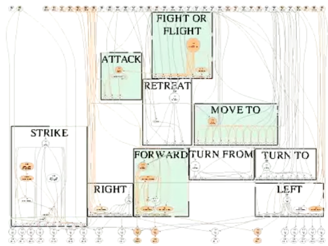
Fight or Flight



- Third level of complexity

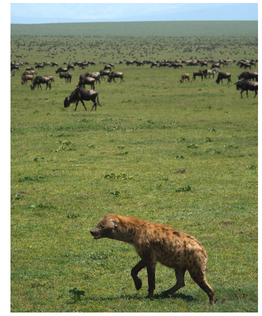
Insight: Body/Brain Coevolution





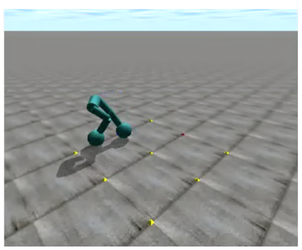
- Evolving body and brain together poses strong constraints
  - Behavior appears believable
  - Worked well also in BotPrize (Turing test for game bots)[64]
- Possible to construct innovative, situated behavior

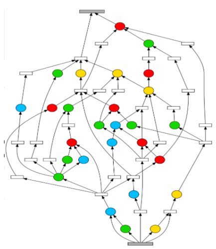
Constructing Intelligent Systems

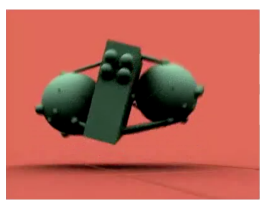


- Believable, complex behavior in embedded environments
  - Open-ended “arms race”
- Similar to self-play e.g. in AlphaGo Zero
  - Complexity beyond human ability to design it
- If we can build open ended environments, should be able to build more complex solutions

Conclusion







- AI extending from prediction to creativity
  - i.e. from modeling to optimization
  - i.e. from Deep Learning to Evolution/RL
- Evolutionary optimization of neural networks can
  - Discover novel and strategic behavior
  - Discover useful complexity for Deep Learning
  - Gain insight into origins of intelligence

Further Material

- www.cs.utexas.edu/users/risto/talks/enn-tutorial
  - Slides and references
  - Demos
  - A step-by-step neuroevolution exercise (evolving behavior in the NERO game)
- www.scholarpedia.org/article/Neuroevolution
  - A short summary of neuroevolution
- www.nature.com/articles/s42256-018-0006-z
  - Nature Machine Intelligence survey on Neuroevolution
- arxiv.org/abs/1902.09635
  - Proposal for NAS benchmark