Evolution of Neural Networks

Risto Miikkulainen
The University of Texas at Austin
and Cognizant AI Labs

Why Use Neural Networks?

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

Why Evolve Neural Networks?

- I. Original role (since 1990s): RL Tasks & especially POMDP
  - Both the structure and the weights evolved (no training)
  - Power from recurrency; behavior
- II. A new role (since 2016): Optimization of Deep Learning Nets
  - Architecture, hyperparameters, functions 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

I. Reinforcement Learning / POMDP 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
Value-Function Reinforcement Learning

- E.g. 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

Policy-Search Reinforcement Learning

- E.g. REINFORCE, policy gradients
  - The policy is optimized directly through hill climbing
  - Works well in simple cases
    - Large/continuous states and actions possible
    - Hidden states (in POMDP) disambiguated through memory
    - Does not scale well

Neuroevolution Reinforcement Learning

- Takes advantage of population-based search
  - In essence, multiple interacting searches
  - Each discover building blocks that are combined
  - Extensive exploration possible
- Makes it possible to scale up:
  - to large spaces (e.g. $2^{25}$ states$^{54}$)
  - to high dimensionality (e.g. up to $1B^{11}$)
  - to deceptive landscapes (with e.g. multiobj and novelty$^{80}$)

How Well Does It Work?

- In the OpenAI Gym CartPole-v0 benchmark vs. PPO, DQN
  - NE converges faster, has lower variance, lower regret
  - NE is more efficient, reliable, and safer$^{16}$
- In a double-pole benchmark vs. Sarsa, Q-MLP, etc.
  - The only method that can find solutions to 1m, 0.1m, POMDP$^{21}$
  - The fundamental difference is exploration
    - Evolution provides more exploration than gradients do$^{32,74,90}$
Neuroevolution for RL/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

Advanced NE 1: Evolving Partial Networks

- Evolving individual neurons to cooperate in networks\(^1,62,65\)
- E.g. Enforced Sub-Populations (ESP\(^19\))
  - Each (hidden) neuron in a separate subpopulation
  - Fully connected; weights of each neuron evolved
- Can be applied at the level of weights, and modules\(^21\)

Why Is It a Good Idea?

- E.g. evolving neurons for robotic control
  - Simulated Kheperas running a maze
- Subpopulations discover & optimize compatible subtasks
  - E.g. slow down with obstacle on front, veer left with obstacle at right, etc.
- Each neuron part of 2-3 subtasks
  - Robust coding of behavior during search

Basic Neuroevolution

- Evolving connection weights in a population of networks\(^61,75,101,102\)
- Chromosomes are strings of connection weights (bits or real)
  - E.g. 10010110101100101111001
  - Usually fully connected, fixed, initially random topology
- A natural mapping between genotype and phenotype
  - GA and NN are a good match!
Advanced NE 2: Evolutionary Strategies

- Evolving complete networks with ES (CMA-ES\textsuperscript{28})
- Small populations, no crossover
- Instead, intelligent mutations
  - Adapt covariance matrix of mutation distribution
  - Take into account correlations between weights
- Why is it a good idea?
  - Discovers good weight combinations → CM

Advanced NE 3: Evolving Network Structure

- Optimizing connection weights and network topology\textsuperscript{2,13,17,103}
- E.g. Neuroevolution of Augmenting Topologies (NEAT\textsuperscript{83,87})
- Based on Complexification
- Of networks:
  - Mutations to add nodes and connections
- Of behavior:
  - Elaborates on earlier behaviors

Why Is It a Good Idea?

- NN search space is complex with nonlinear interactions
- Complexification keeps the search tractable
  - Start simple, add more sophistication
- Incremental discovery of complex solutions

Advanced NE 4: Indirect Encodings (1)

- Instructions for constructing the network evolved
  - Instead of specifying each unit and connection\textsuperscript{2,13,60,81,103}
- E.g. Cellular Encoding (CE\textsuperscript{25})
- 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)
- Evolve a neural network (CPPN) to generate spatial patterns
  - 2D CPPN: \((x, y)\) input \(\rightarrow\) grayscale output
  - 4D CPPN: \((x_1, y_1, x_2, y_2)\) input \(\rightarrow\) \(w\) output
  - Connectivity and weights can be evolved indirectly
  - Works with very large networks (millions of connections)

Why Is It a Good Idea?

- Describes structure efficiently
  - Recurrency symbol in CE: XOR \(\rightarrow\) parity
  - Repetition with variation in CPPNs
- Useful for evolving topology
  - E.g. large structured networks
  - E.g. repetition of motifs

Future Opportunities

- Several possible directions
  - More general L-systems; developmental codings; embryogeny
  - Scaling up spatial coding
  - Genetic Regulatory Networks
  - Evolution of symmetries
- Theory starting to emerge
  - Expressive Encodings: Simple GAs are universal probability approximators

Further NE Techniques

- Incremental and multiobjective evolution
- Utilizing population culture
- Utilizing evaluation history
- Evolving NN ensembles and modules
- Evolving transfer functions and learning rules
- Bilevel optimization of NE
- Evolving LSTMs for strategic behavior
- Extrapolation with Context+Skill modules
- Combining learning and evolution
- Evolving for novelty
Evolving for Novelty

Motivated by humans as fitness functions
- E.g. picbreeder.com, endlessforms.com
- CPPNs evolved; Human users select parents

No specific goal
- Interesting solutions preferred
- Similar to biological evolution?

Novelty Search

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

Novelty Search Demo (1)

Illustration of stepping stones
- 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 (2)

Fitness-based evolution is rigid
- Requires gradual progress
- Novelty-based evolution is more innovative, natural
- Allows building on stepping stones
- How to guide novelty search towards useful solutions?
- Quality Diversity methods

DEMO
### Neuroevolution Applications

<table>
<thead>
<tr>
<th>Control</th>
<th>Pole-Balancing</th>
<th>Satellite Ass.</th>
<th>Helicopter</th>
<th>Rocket</th>
</tr>
</thead>
<tbody>
<tr>
<td>Robotics</td>
<td>Soccer</td>
<td>Driving</td>
<td>Bipedal</td>
<td>Multi-legged</td>
</tr>
<tr>
<td>Games</td>
<td>Othello</td>
<td>NERO</td>
<td>Pac-Man</td>
<td>Unreal</td>
</tr>
<tr>
<td>Alife</td>
<td>Duel</td>
<td>Predators</td>
<td>Hyenas/Zebraf</td>
<td>Virtual Creatures</td>
</tr>
</tbody>
</table>

### Example 1: 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

- Wandering, unstuck etc. based on scripts & learning from humans
- Evolve effective fighting behavior
- Persistent gap: 30% vs. 80% human
  - Evolving to win results in unnatural behaviors
  - Human judges do not understand their expertise

### After Five Years, Success!!!

- Human-like behavior with resource limitations (speed, accuracy...)
  - Best bot better than 50% of the humans
  - Two teams human 50% of the time
- Fascinating challenges remain:
  - Judges can still differentiate in seconds
  - Judges lay cognitive, high-level traps
  - Team competition: collaboration as well
- DEMO
Example 2: Optimizing COVID-19 NPIs

Train a NN to predict COVID-19 cases
- Based on number of cases in different countries over time
- And non-pharmaceutical interventions (NPIs) over time

Using the predictive model as a surrogate, evolve a NN to recommend NPIs
- Resulting in smallest number of cases
- With minimal economic cost

Not just what will happen, but what we should do about it!

COVID-19 Predictions and Prescriptions

Retrained daily since May 2020
- Based on data from Oxford University
- Adapting to the different stages of the pandemic
- Generalizing from experiences across the world

Recommendations about two weeks in advance, e.g.
- May 2020: Focus on schools and workplaces (i.e. indoors)
- Sept 2020: Focus on gatherings, travel restrictions
- March 2021: Delta surge: India lockdown
- Dec 2021: Missed omicron surge; everywhere at once
- March 2022: Masking to avoid a second omicron surge

Interactive demo:
- https://evolution.ml/demos/npidashboard

Using Neuroevolution to Leverage Human Insight

XPRIZE Pandemic Response Challenge 2021
- 169 expert-designed prescriptors
- Distill into neural networks and evolve further
- Improve upon expert-designed entries
- Improve upon evolution from scratch
- Can realize latent potential hidden in poor entries
- Technology to bring the community effort together

Part I Conclusion: Neuroevolution RL

- A powerful way to train networks when gradients not available
  - E.g. recurrency in POMDP domains
- Many evolutionary techniques are a good match with NE
  - Partial solutions, CMA, Complexification, Indirect, Novelty, Constrained
- Can discover surprising, believable, effective solutions
# Optimization of Deep Learning Systems

Deep learning systems operate at a much larger scale:
- $10^8 - 10^{12}$ parameters
- Overparameterized; trained by gradient descent

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?

## Configuring Deep Learning with Evolution

(A) Fundamental: Neural Architecture Search
- Optimizing structure and hyperparameters
- Takes advantage of exploration in EC

(B) Extended: Data and training
- Loss functions, activation functions, data augmentation, initialization, learning algorithm
- Takes advantage of flexibility of EC

## Evolutionary NAS

Evolution is a natural fit:
- Population-based search covers the space
- Crossover between structures discovers principles

Moreover,
- Can build on Neuroevolution work since the 1990s:
  - partial solutions, complexity, indirect encoding, novelty search
- Applies to continuous values; discrete choices; graph structures; combinations
- Can evolve hyperparameters; nodes; modules; topologies; multiple tasks
**E.G. NAS with CoDeepNEAT**

- Evolution at three levels:
  - Module subpopulations optimize building blocks
  - Blueprint population optimizes their combinations
  - Hyperparameter evolution optimizes their instantiation

- Fitness of the complete network drives evolution
  - Candidates need to be evaluated through training
  - Expensive; use partial training, surrogates...

---

**Making NAS Evaluations Practical**

- Population-based training (DeepMind, Cognizant)
  - Continual training and evolution

- NAS benchmarks created to help evaluate (Google, Baidu, Freiburg)
  - Collections of known architecture evaluations, surrogates

- Scaling and regularization (Google, Uber)
  - State-of-the-art at the time in CIFAR-10, CIFAR-100, ImageNet

- Specialized crossover operators (Cognizant)

---

**Optimizing Other Aspects of Deep Learning Design**

- Optimizing activation functions and loss functions (Cognizant)
  - Regularization and refinement
  - Designing machine learning algorithms with GP (Google)

- Coevolution of multiple aspects of network design?

---

**Evolutionary AutoML**

- Current AutoML: Mostly hyperparameter optimization
- Future Evolutionary AutoML: Many design aspects

- Performance
  1. Improve state of the art
     - With sufficient compute
  2. Improve over naive baseline
     - Service makes broadly available

- Applicability
  3. Minimize network resources
     - Train and run networks faster
  4. Extend small datasets
     - Multitasking with related datasets
1 & 2 in Evolving Age-Estimation Networks

<p>|</p>
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Possible Values</th>
<th>Type</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>Algorithm</td>
<td>[adam, rmsprop]</td>
<td>Enum</td>
<td>Opt</td>
</tr>
<tr>
<td>Initial Learning Rate (LR)</td>
<td>[1e-5, 1e-1]</td>
<td>Float</td>
<td>Opt</td>
</tr>
<tr>
<td>Momentum</td>
<td>[0.7, 0.99]</td>
<td>Float</td>
<td>Opt</td>
</tr>
<tr>
<td>(Weight Decay) / LR [80]</td>
<td>[1e-7, 1e-1]</td>
<td>Float</td>
<td>Opt</td>
</tr>
<tr>
<td>Patience (Epochs)</td>
<td>[1, 30]</td>
<td>Int</td>
<td>Opt</td>
</tr>
<tr>
<td>SWA Epochs</td>
<td>[1, 20]</td>
<td>Int</td>
<td>Opt</td>
</tr>
<tr>
<td>Rotation Range (Degrees)</td>
<td>[0, 360]</td>
<td>Int</td>
<td>Aug</td>
</tr>
<tr>
<td>Width Shift Range</td>
<td>[0.01, 0.3]</td>
<td>Float</td>
<td>Aug</td>
</tr>
<tr>
<td>Height Shift Range</td>
<td>[0.01, 0.3]</td>
<td>Float</td>
<td>Aug</td>
</tr>
<tr>
<td>Shear Range</td>
<td>[0.01, 0.3]</td>
<td>Float</td>
<td>Aug</td>
</tr>
<tr>
<td>Zoom Range</td>
<td>[0.01, 0.3]</td>
<td>Float</td>
<td>Aug</td>
</tr>
<tr>
<td>Horizontal Flip</td>
<td>True, False</td>
<td>Bool</td>
<td>Aug</td>
</tr>
<tr>
<td>Vertical Flip</td>
<td>True, False</td>
<td>Bool</td>
<td>Aug</td>
</tr>
<tr>
<td>Cutout Probability [?]</td>
<td>[0.01, 0.99]</td>
<td>Float</td>
<td>Aug</td>
</tr>
<tr>
<td>Cutout Max Proportion [?]</td>
<td>[0.05, 0.3]</td>
<td>Float</td>
<td>Aug</td>
</tr>
<tr>
<td>Base Model Output Blocks</td>
<td>[B1, B2, B3]</td>
<td>Subset</td>
<td>Arch</td>
</tr>
<tr>
<td>Loss function ( \lambda ) in Eq. 5</td>
<td>[0.1]</td>
<td>Float</td>
<td>Arch</td>
</tr>
</tbody>
</table>

Estimate age from a facial image

Evolving multiple design aspects
- Learning, data augmentation hyperparameters
- Seeded architecture search
- Loss-function optimization: Combination of MAE and CE

Also
- Population-based training
- Ensembling of evolved solutions

3. 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

Could we optimize for size directly?

Age-Estimation Results

- D0 stages: ResNet-50, DenseNet-121
- D1 stages: DenseNet-169, DenseNet-201, more epochs, EfficientNet-B6, ensembling
- Human optimization of ResNet-50 (D0), EfficientNet-B6 (D1)

Evolution improves significantly over SotA image models
- Fit the design to the task
- Optimizes better than humans can
  - Many more parameters simultaneously
- Performance exceeds that of humans: 2.19 vs. 3-4 years

3. Minimize Network Resources

- 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
- Evolution can find solutions that fit design constraints
4. Extend Small Datasets

Recognize handwritten characters in a given alphabet
Not enough samples to learn well
• A common problem in deep learning
Could we learn from multiple alphabets at once?
• More generalizable embeddings
• Can learn each task better
Evolve architecture to combine multiple tasks
• Network architecture can have a large effect
• A good domain for NAS

Future: Large Language Models

E.g. Evolutionary prompting for NAS (EvoPrompting)
• Existing architectures as prompts; generate new
• Tune the prompts based on performance
Better evolution through LLMs?
• Evolution through large models (ELM)
• Language model crossover (LMX)
Better LLMs through evolution?
• Evolutionary NAS, fine tuning, alignment, prompting?

Multitasking Benchmarks

State-of-the-art in two ML benchmarks:
• Omniplot multialphabet character recognition
  • Improved state-of-the-art 31%
  • Demo: evolution.ml/demos/omnidraw
• CelebA multiattribute face classification
  • Improved state-of-the-art 0.75%
  • Demo: evolution.ml/demos/celebmatch
Improves learning in each task
• Even when little data available
Extend small datasets with multiple tasks

Part II Conclusion: Optimizing Deep Learning Designs

• Deep learning designs are too complex for humans to optimize
• Evolutionary techniques are a good fit
  • Large, structured space; continuous, discrete, and structured
• Can be applied to multiple aspects of the design
  • How to utilize their interactions?
  • How to evaluate candidates efficiently?
III. Emergence of Intelligence

- Origins of intelligence: Embodied optimization
- Body-Brain Coevolution 35,36,82
  - Body: Blocks, muscles, joints, sensors
  - Brain: A neural network (with general nodes)
  - Evolved together in a physical simulation

Encapsulation

- Once evolved, a trigger node is added
- 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)\(^\text{77}\)
- Possible to construct innovative, situated behavior

Constructing Intelligent Systems

- Believable, complex behavior in embedded environments
  - Open-ended “arms race”\(^\text{68}\)
  - Similar to self-play e.g. in AlphaGo Zero
    - Complexity beyond human ability to design it
  - If we can build open ended environments, we should be able to build more complex solutions
    - Co-evolve environments and behaviors? (e.g. POET\(^\text{98}\), EUREQA\(^\text{76}\))
  - Challenge: Establish major transitions\(^\text{56}\)
Conclusion

- Neuroevolution is a powerful approach for POMDPs
  - Discovers surprising, believable, effective behavior
- Games, robotics, control, alfie, decision-making...
- Makes complex DL architectures possible
  - Structure, components, hyperparameters, etc. fit to the task
- Automatic design of learning machines
- A possible future focus: Emergence of intelligence
  - Body/brain co-evolution; Competitive co-evolution
  - Evolution of memory, language, learning; AGI

Further Material

- Neuroevolution sessions at GECCO!
  - 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)
  - nn.cs.utexas.edu/?miikkulainen:encyclopedia20-ne
- A short summary of neuroevolution
  - www.nature.com/articles/s42256-018-0006-z
  - Nature Machine Intelligence survey on Neuroevolution

References I


References II

References III


References IV


References V


References VI


References VII


References VIII


References IX


References X