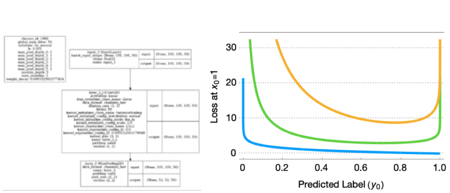
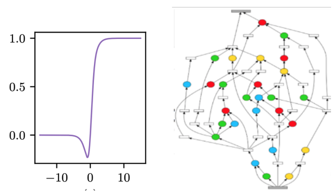
Can Different Metalearning Methods be Combined Synergetically?

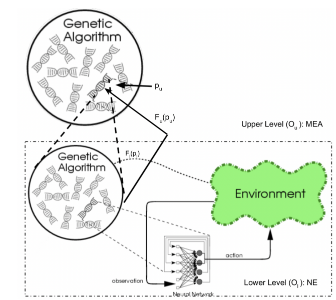
- Optimization of multiple design aspects should enhance performance.
- Taking advantage of potential synergies.
- Potential for taking advantage of complexity beyond human design.





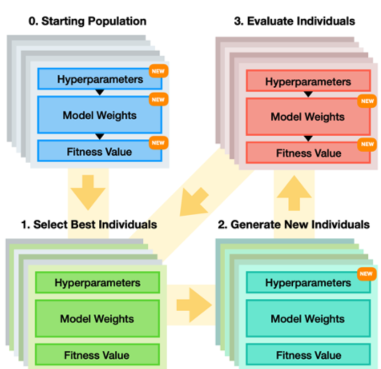
Challenges in Multi-Aspect Evolution

- Searching all design aspects simultaneously is computationally prohibitive.
- Full inner-outer loop structures would be too costly.
- Solution: Use surrogate models and alternate evolutionary focus.



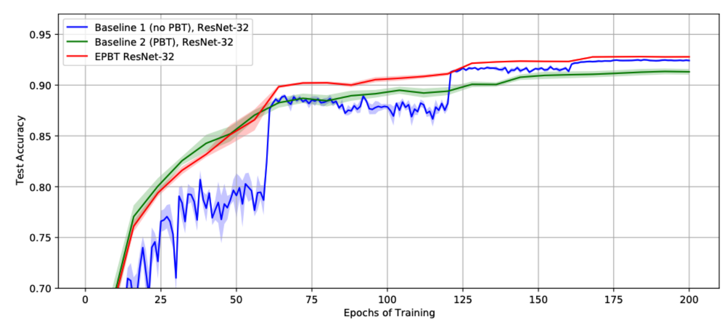
EPBT System for Synergistic Evolution

- EPBT combines hyperparameter tuning, loss function optimization, and population-based training.
- Evolves hyperparameters and loss functions during training.
- Overfitting becomes a problem:
  - Use novelty pulsation to prevent convergence.
  - Learn from labels+best individuals to regularize.



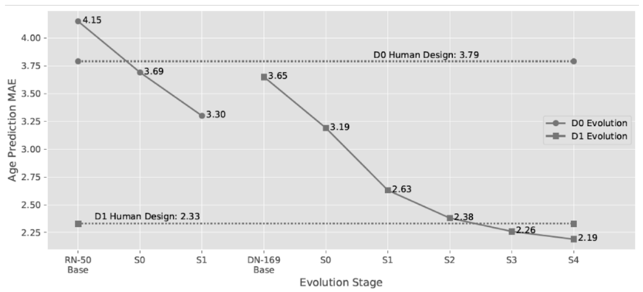
EPBT System for Synergistic Evolution

- Successful in e.g. CIFAR-10 over baseline and PBT.
- Smooth improvement instead of jumps.
- Difficult to get synergies to emerge.
- Is it worth it?



A Natural Experiment: Human Design vs. Evolutionary Metalearning

- Age estimation model design simultaneously by humans vs. metalearning.
- Over the same time period, utilizing the same base technologies.
- A friendly but real competition.



Medical Aesthetics

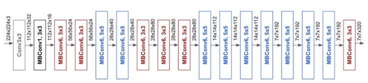
>>>>

A family of treatments to improve a patient's appearance
- E.g. Alter facial skin texture through Botox or filler injections
- Outcome difficult to measure, often subjective

One potentially measurable goal is to reduce perceived age
- Can we make it quantitative through AI-based age estimation?

Cognizant

Approach



Starting point: A neural network trained in age estimation
- E.g. DenseNet, EfficientNet; Celebrity dataset

Improve performance by
- Utilizing a dataset of actual patient populations
- Optimizing the neural network through evolution

Demonstrate that treatment significantly reduces age estimates vs. placebo
- Measure confidence in the predictions using RIO:
  A Gaussian Process model of residual error with input/output kernel

Cognizant

Age-Estimation Datasets

IMDB dataset of 172,000 celebrity faces commonly used
- Often retouched images, or treatments already done
- Difficult to estimate age
- E.g. DenseNet-121 validation error 7.43 years

Collected two patient datasets (with different treatments)
- D0: 3719 patients, ages 18-79, 10,837 training, 2692 testing images, 224x224
- D1: 5998 patients, ages 18-80, 18,537 training, 3733 testing images, 512x512
- E.g. DenseNet-169 validation error 3.65 years

Thus, using realistic datasets matters

Cognizant

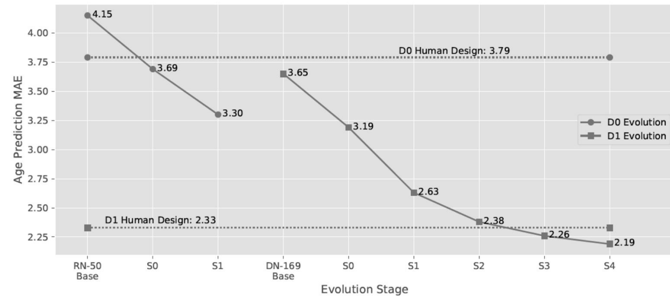
Evolving Age-Estimation Networks

Parameter	Possible Values	Type	Class
Algorithm	[adam, rmsprop]	Enum	Opt
Initial Learning Rate (LR)	[1e-5, 1e-3]	Float	Opt
Momentum	[0.7, 0.99]	Float	Opt
(Weight Decay) / LR [26]	[1e-7, 1e-3]	Float	Opt
Patience (Epochs)	[1, 20]	Int	Opt
SWA Epochs [21]	[1, 20]	Int	Opt
Rotation Range (Degrees)	[1, 60]	Int	Aug
Width Shift Range	[0.01, 0.3]	Float	Aug
Height Shift Range	[0.01, 0.3]	Float	Aug
Shear Range	[0.01, 0.3]	Float	Aug
Zoom Range	[0.01, 0.3]	Float	Aug
Horizontal Flip	{True, False}	Bool	Aug
Vertical Flip	{True, False}	Bool	Aug
Cutout Probability [7]	[0.01, 0.999]	Float	Aug
Cutout Max Proportion [7]	[0.05, 0.5]	Float	Aug
Pretrained Base Model	Keras App. [5]	Enum	Arch
Base Model Output Blocks	{B0, B1, B2, B3}	Subset	Arch
Loss function λ in Eq. 5	[0, 1]	Float	Arch

Evolving solutions using LEAF
- Evolve backprop, data augmentation, architecture hyperparameters
- Population-based training:
  20 epochs in each generation
- Loss-function optimization
- Ensembling of evolved solutions

Fitness, training loss a combination of
- Minimize Mean Absolute Error (MAE)
- Cross-entropy (CE)

Age-Estimation Results

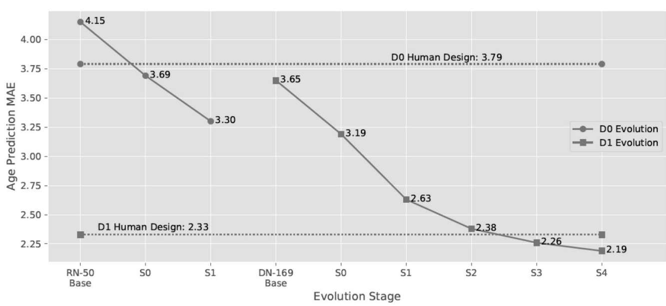


- D0 stages:
  ResNet-50,
  DenseNet-169
- D1 stages:
  DenseNet-169,
  DenseNet-201,
  EfficientNet-B6,
  epochs, resolution
- Human optimization based on ResNet-50, EfficentNet-B6

Evolution improves significantly over SotA image models
- Fit to the design to the task

Optimizes better than humans can
- Many more parameters simultaneously

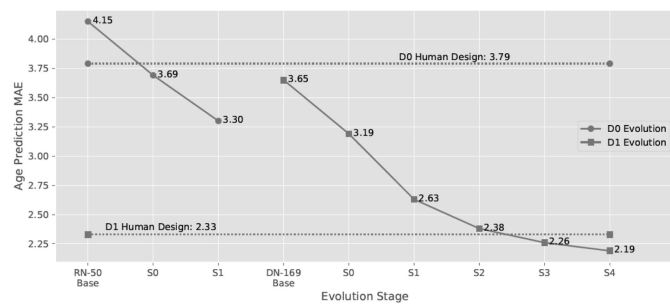
Age-Estimation Discoveries



Meaningful data augmentation
- Vertical flips instead of horizontal: images had 90-degree rotation
- 5x width shift range: Less overfitting to forehead and chin

Different losses at different stages: Less overfitting with MAE early

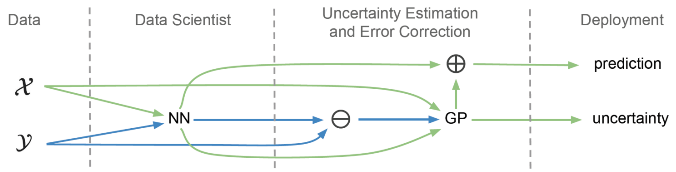
Evaluating Treatments



Performance exceeds that of humans: 2.19 vs. 3-4 years
- A possible basis for quantitative evaluation of treatment effects

Need a method to estimate confidence in the predictions

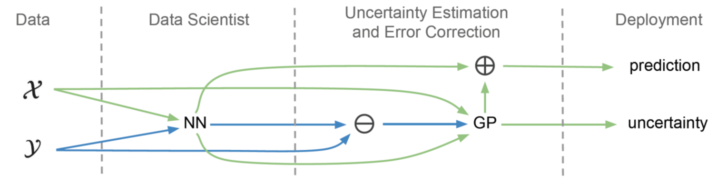
RIO: Residual Estimation with an Input+Output Kernel



(Qiu et al. 2020)

- Adds to existing NNs: No changes to structure or pipeline
  - Based on modeling prediction residuals with GP
  - Includes both NN input and output as kernel
- Estimates uncertainty and improves predictions

Why Does RIO Work?



- Why is RIO better than NN alone or GP alone?

- NN is expressive (i.e. has high variance)
  - Learns structure that GP would treat as noise
- Remaining structure is easier to learn
  - GP can capture part of it
  - GP is more regular than NN

Treatment Evaluation Dataset (D2)

A single Botox study with 787 patients 21-76 years
- 3925 images taken before treatment
- 68,799 after at 1 and 2 weeks, monthly until 6 months

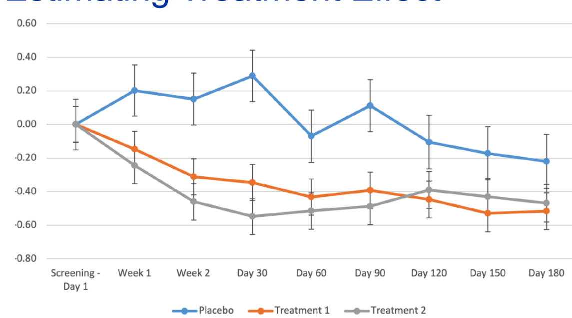
Two different treatments (injection volumes)
- 156 placebo patients; 5190 images
- Single injection only

Pre-treatment age bias removed

RIO evaluated with pre-treatment data
- Improved age estimation from 1.61 to 1.48 years
- Accurate coverage of 95%, 90%, 68% confidence intervals

Metric	Value
Original MAE	1.61
MAE with RIO	1.48
95% CI Coverage	94.2%
90% CI Coverage	89.2%
68% CI Coverage	69.2%

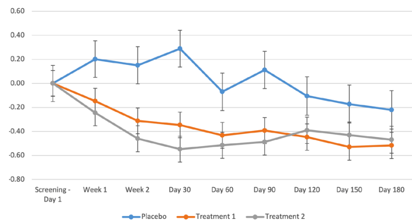
Estimating Treatment Effect



Treatment reduces perceived age significantly compared to placebo injections
- 0.5 years in 6 months, i.e. 1 year overall
- Main effect in 1-2 months, then stable
- Actual treatments include multiple injections, with a cumulative effect

A new role for AI: Make subjective evaluations quantitative.

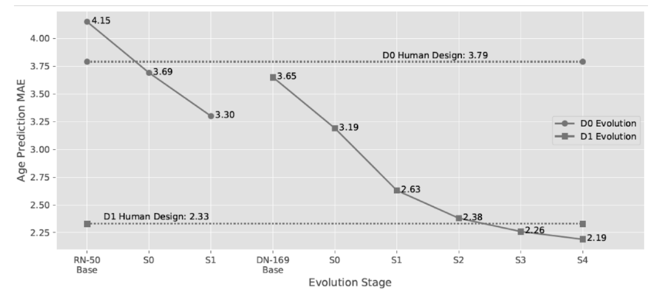
Future Work



Evaluate cumulative effect of multiple injections, other treatments, combinations
Evaluate other outcomes, e.g. natural look (with GANs)
Predict the effect of treatments
Optimize the treatments to maximize desired outcomes

Conclusion of Synergistic Metalearning

- Evolutionary metalearning can outperform human optimization by leveraging synergies.
- Combined methods allow exploration of design spaces beyond human capabilities.
- It is difficult to get to work, but it is worth it.

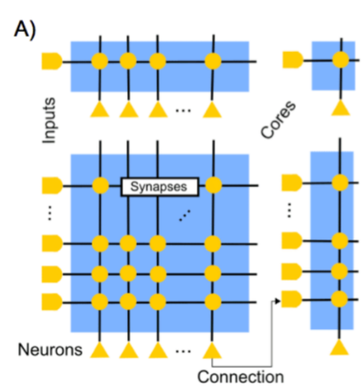


Conclusion: The Power and Potential of Metalearning in Neuroevolution

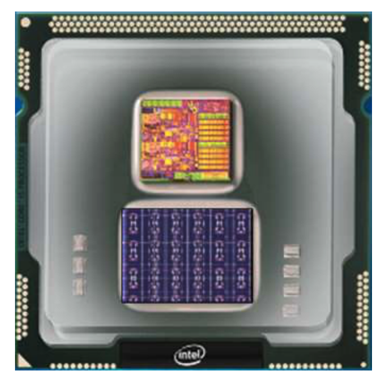
- **Purpose of Metalearning:**
  - Metalearning makes neural network designs automatic, improving upon human design.
  - Can evolve to take advantage of customized designs for specific settings.
- **Key Successes:**
  - Improvement neuroevolution through bilevel optimization.
  - Discovery of regularization through Baikal loss function.
  - Customization through evolved activation functions and data augmentation.
  - Effective synergetic metalearning, and discovery of learning methods.
  - Demonstrated competitive edge over human-designed models in age estimation.
- **Future Opportunities:**
  - Explore synergies between more complex aspects, like architecture and learning method evolution.
  - Refine surrogate modeling to expand search spaces further without increased computational cost.
  - Extend metalearning to newer architectures, such as transformers and diffusion models.

Introduction to Neuromorphic Systems

- Neuromorphic computing: Hardware for spiking neural networks.
- Notable implementations: IBM's TrueNorth and Intel's Loihi, with 1M spiking neurons.



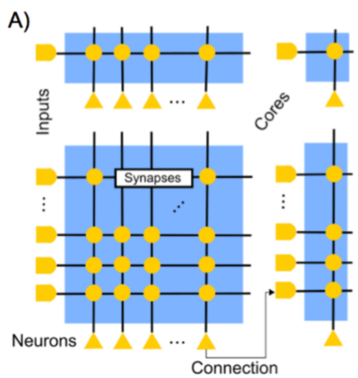
IBM TrueNorth



Intel Loihi

Motivations for Neuromorphic Computing

- Primary goal: Energy-efficiency.
- Also fault-tolerance, real-time computing, and compact designs.

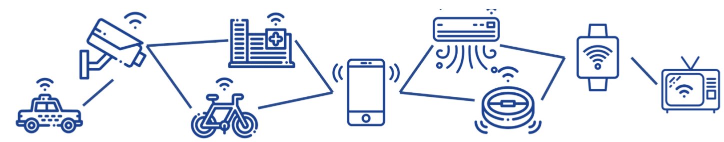


IBM TrueNorth


Intel Loihi

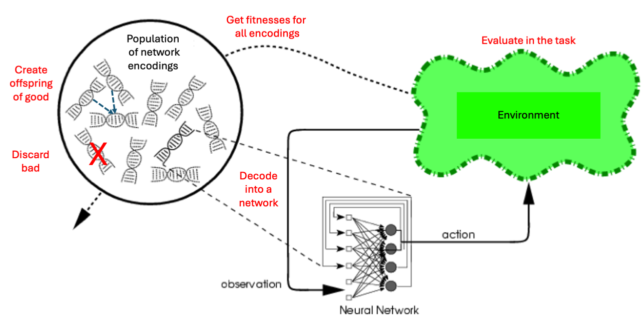
Potential Applications for Neuromorphic Systems

- Suitable for vision, sensing, control, and low-power devices.
- Edge applications: auditory/visual detection, brain-machine interfaces.
- Neuromorphic systems offer feasible low-power solutions for remote applications.



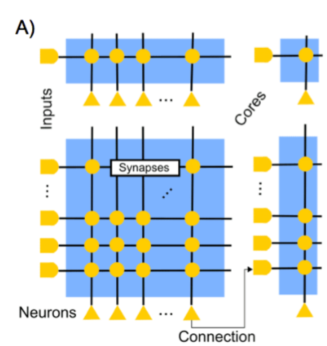
Why Use Neuroevolution for Neuromorphic Design?

- Bypasses the need for gradients (hard to compute in hardware).
- Takes advantage of small networks (easy to manufacture)
- Arbitrary connectivity, recurrency.
- Many hyperparameters; conntinuous, discrete, binary, structure.

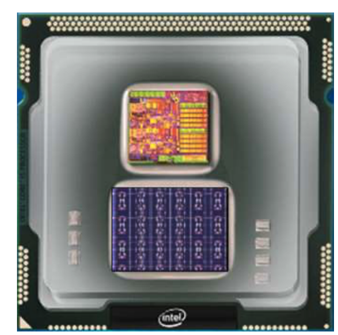


Challenges and Opportunities

- Different setting from standard neuroevolution.
  - Spike timing, interference, leaky integration, refractory periods, low precision.
  - Main goal is not accuracy, but energy efficiency.
  - Many secondary objectives.
- Optimizations matter!
  - Often qualitative jumps result from changes in structure
  - Principles not known.
  - Many secondary objectives.
- Potential to co-design hardware and algorithms, optimizing both.

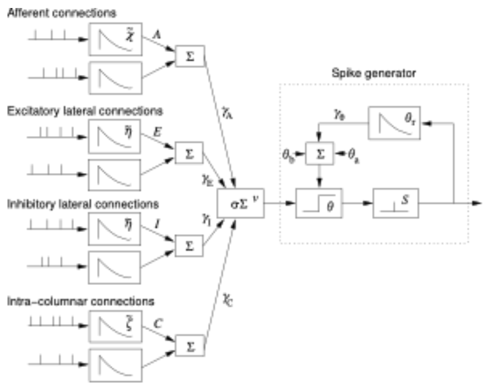


IBM TrueNorth


Intel Loihi

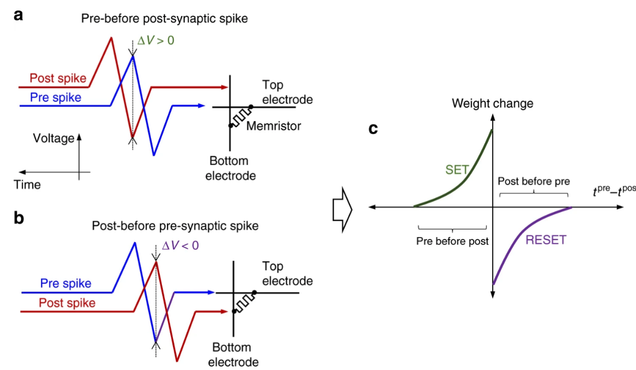
Spiking Neurons and Hardware Implementations

- Spiking neurons use discrete events, reducing power usage.
- Offers new approaches to emulate biological neural networks.
- Is learning possible as well?

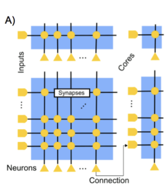


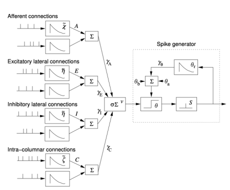
Learning with Spike-Timing-Dependent Plasticity (STDP)

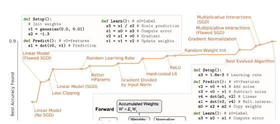
- STDP strengthens connections when presynaptic spikes precede postsynaptic firing.
- Encourages unsupervised learning based on timing.
- Extends Hebbian principle: "neurons that fire together wire together."



Opportunity: Optimize Neuromorphic Learning





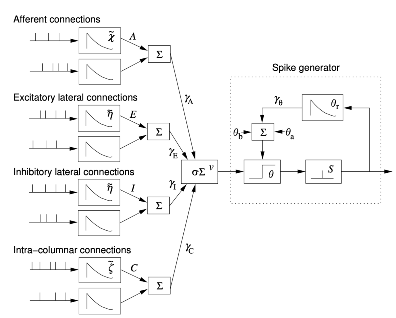


Hardware exists that allows modifying learning rules
- Should take advantage of it

Hunch: there are principles like STDP that can be discovered
- A big idea in the long term: modify hardware to fit
- Maybe insights from neuroscience?

The main goal is performance but also power consumption (as always)

Details of Neuromorphic Learning

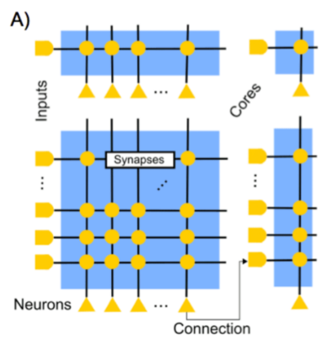


Accessible input variables:
- Pre and postsynaptic spikes and their timing
- Possibly others as well, especially on a simulator
  - Spike timings in a neighborhood
  - Spike timings through history
  - State of the neuron, other global state descriptions
- Gradients usually are not available

Output variables
- Learning rule parameters
- Shape of the function: GP, Taylor

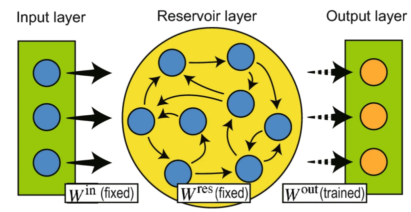
Opportunity: Optimize Network Design

- Networks with unreliable devices (memristors)
  - Can we not only mitigate, but actually leverage their behaviors and interactions?
  - Similar to the magnetic flux effect in FPGAs (Thompson 1998)
- Need good simulators because we need principled noise and interactions.
- Or develop a good surrogate model based on experiments.
- Start with an initial architecture and place devices into it.



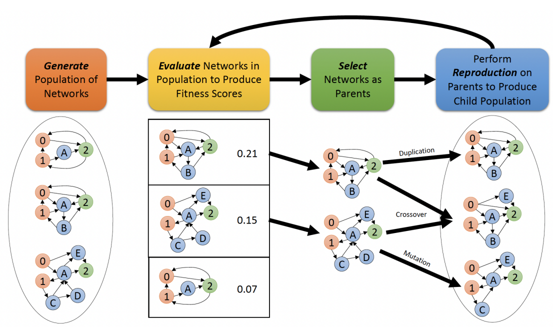
Initial Approach: Reservoir Computing

- Reservoir networks with random recurrent connectivity create sequences.
- Train a network on top to take advantage of them.
- Effective for tasks requiring continuous temporal processing.
- Could also optimize the reservoir with neuroevolution.



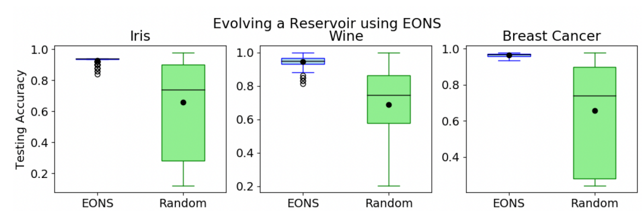
EONS Framework for Neuromorphic Evolution

- Evolutionary Optimization of Neuromorphic Systems (EONS): flexible structure and parameter optimization.
- Adapts to hardware constraints and task requirements.
- Allows for hardware-based implementation or simulation.



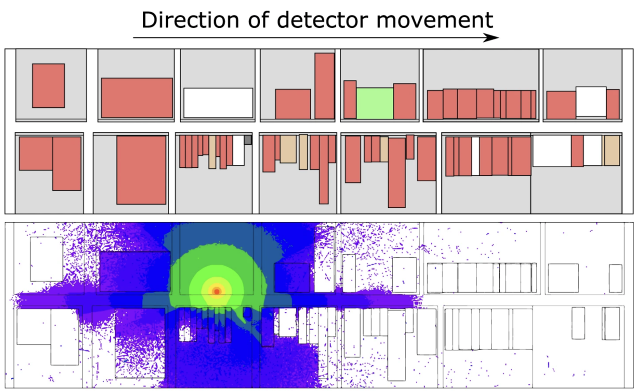
Optimizing Reservoir Architectures with EONS

- EONS optimizes reservoir hyperparameters, connectivity, and weights.
- Used in applications requiring continuous learning and adaptability.
- Enhanced performance on complex tasks compared to grid search methods.



Case Study: Radiation Anomaly Detection

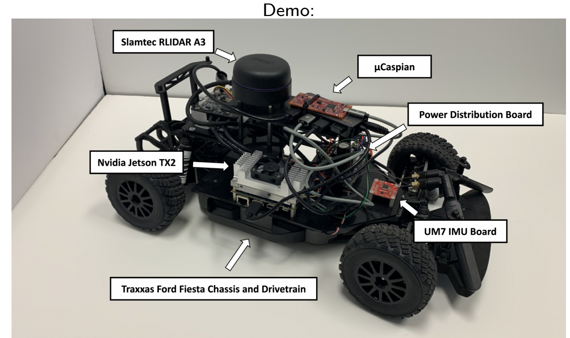
- ORNL dataset: A detector moving in an urban environment to find nuclear threats.
- Detects hidden gamma-ray sources with low power consumption.
- EONS optimizes network topology, encoding, and spiking thresholds.
- Achieved competitive sensitivity with significant energy savings.



Case Study: Control of Autonomous Vehicles (F1Tenth)

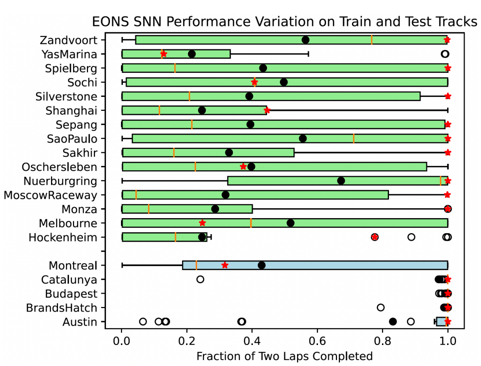
- Neuroevolution used for low-power autonomous control in the F1Tenth race car.
- Optimized controller ran on μCaspian neuromorphic board.

Demo:



Neuroevolved Controller Performance

- Trained on five tracks, tested on 15 others (in simulation)
- Evolved controllers showed robust performance across diverse environments (best controller on average indicated by the red star).
- Smaller, energy-efficient designs with better results than human-tuned controllers.
- Demonstrated performance transfer to the physical car as well.



Conclusions on Neuromorphic Neuroevolution

- **Why Neuromorphic Neuroevolution?**
  - Optimizes neural architectures for edge applications.
  - Improves energy use, size, fault-tolerance, latency.
- **Key Successes:**
  - Improved performance on classification, detection, and control tasks with minimal energy consumption.
  - E.g. radiation anomaly detection with low-power.
  - E.g. controller for an autonomous F1Tenth vehicle.
- **Future Directions:**
  - Development of co-evolution techniques for hardware and neural architectures.
  - Integration of novel learning mechanisms, possibly advancing beyond current models like STDP.
  - Exploring new edge applications.