Weight Agnostic Neural Networks

(To Appear at NeurIPS 2019)

Adam Gaier, David Ha

Motivations

Today's Deep Learning:

- **Design the neural network <u>architecture</u>** (we believe to be *suitable* for encoding a policy for the task. E.g., CNNs, LSTMs, self-attention….)

- **Find the <u>weight parameters</u>** (for the fixed architecture, solve the *optimization* problem, using back-propagation.)

Motivations

Today's Deep Learning:

- **Design the neural network <u>architecture</u>** (we believe to be *suitable* for encoding a policy for the task. E.g., CNNs, LSTMs, self-attention….)

- **Find the <u>weight parameters</u>** (for the fixed architecture, solve the *optimization* problem, using back-propagation.)

<u>Question:</u> Can we find neural **architectures** which are naturally capable of performing a task even when their **weight parameters** are randomly sampled (i.e., no learning at all)?

Motivations

<u>Question:</u> Can we find neural **architectures** which are naturally capable of performing a task even when their **weight parameters** are randomly sampled (i.e., no learning at all)?

Motivations

<u>Question:</u> Can we find neural **architectures** which are naturally capable of performing a task even when their **weight parameters** are randomly sampled (i.e., no learning at all)?

Bio-plausibility (precocial behaviors):

- **Innate ability** (e.g. lizard 🦎 / snake 🐍 hatchlings can escape from predators..)
- **Few-shot learning** (e.g. ducks 🦆 are able to swim and eat, turkeys 🦃 can visually recognize predators..)

Motivations

<u>Question:</u> Can we find neural **architectures** which are naturally capable of performing a task even when their **weight parameters** are randomly sampled (i.e., no learning at all)?

Deep learning building blocks (inductive biases):

- **CNN, RNN, LSTM, self-attention, capsule…**

- **Randomly-initialized CNNs** can be used effectively for image processing tasks such as **superresolution**, **inpainting** and **style transfer**.

- **A randomly-initialized LSTM** with a learned linear output layer can predict time series.

Related work

<u>Neural Architecture Search (NAS):</u>

- Search neural network topology using **evolutionary algorithms** (e.g., NEAT algorithm)

- Narrow the search space to architectures composed of **basic building blocks** (CNN/RNN cell/self-attention)

- Time-costly **inner loops** for training (find the optimal weight parameters)

- Never claimed that the solution is innate to the structure of the network. — The **weights are the solution**; the found architectures only a substrate for the weights to inhabit.

Related work

Neural Architecture Search (NAS):

- **S** • Search neural network topology using **evolutionary algorithms** (e.g., NEAT algorithm)

- **D** • Narrow the search space to architectures composed of **basic building blocks** (CNN/RNN cell/self-attention)

- **D** • Time-costly **inner loops** for training (find the optimal weight parameters)

- **D** • Never claimed that the solution is innate to the structure of the network. — The **weights are the solution**; the found architectures only a substrate for the weights to inhabit.

Related work

Bayesian Neural Networks (BNN):

- Weight parameters are **sampled** from a distribution.
- The distribution is **learned** and usually has more parameters then the number of weights.
- Recent **variance networks** shows that network ensembles whose weights are sampled from a **zero mean** distribution can perform well on image recognition tasks (counter-intuitive…)

Related work

Bayesian Neural Networks (BNN):

S • Weight parameters are **sampled** from a distribution.

D • The distribution is **learned** and usually has more parameters then the number of weights.

S • Recent **variance networks** shows that network ensembles whose weights are sampled from a **zero mean** distribution can perform well on image recognition tasks (counter-intuitive…)

Related work

<u>Algorithmic Information Theory (AIT):</u>

- Occam's razor. Simplifying the search space of its **weights**. (soft-weight sharing, regularization, etc.)

- Recent work establishes the description length of deep learning models based on **architectures**. Goal of this paper is to fined minimal architectures, instead of simple weights.

Related work

Network Pruning:

- **Pruned networks** that can achieve image classification accuracies that are much **better than chance** even with randomly initialized weights.

- **Complementary approach**: starts with a full, trained network, and takes away connections.

Related work

Connectomics:

- **Wiring diagram** of all neural connections of the brain. Human connectome has ~90 billion neurons and ~150 trillion synapses.

- Hope to gain insight about how the brain learns and represents memories in its **connections**.

- **Deemphasize** learning of weight parameters to test the importance of the network architecture.

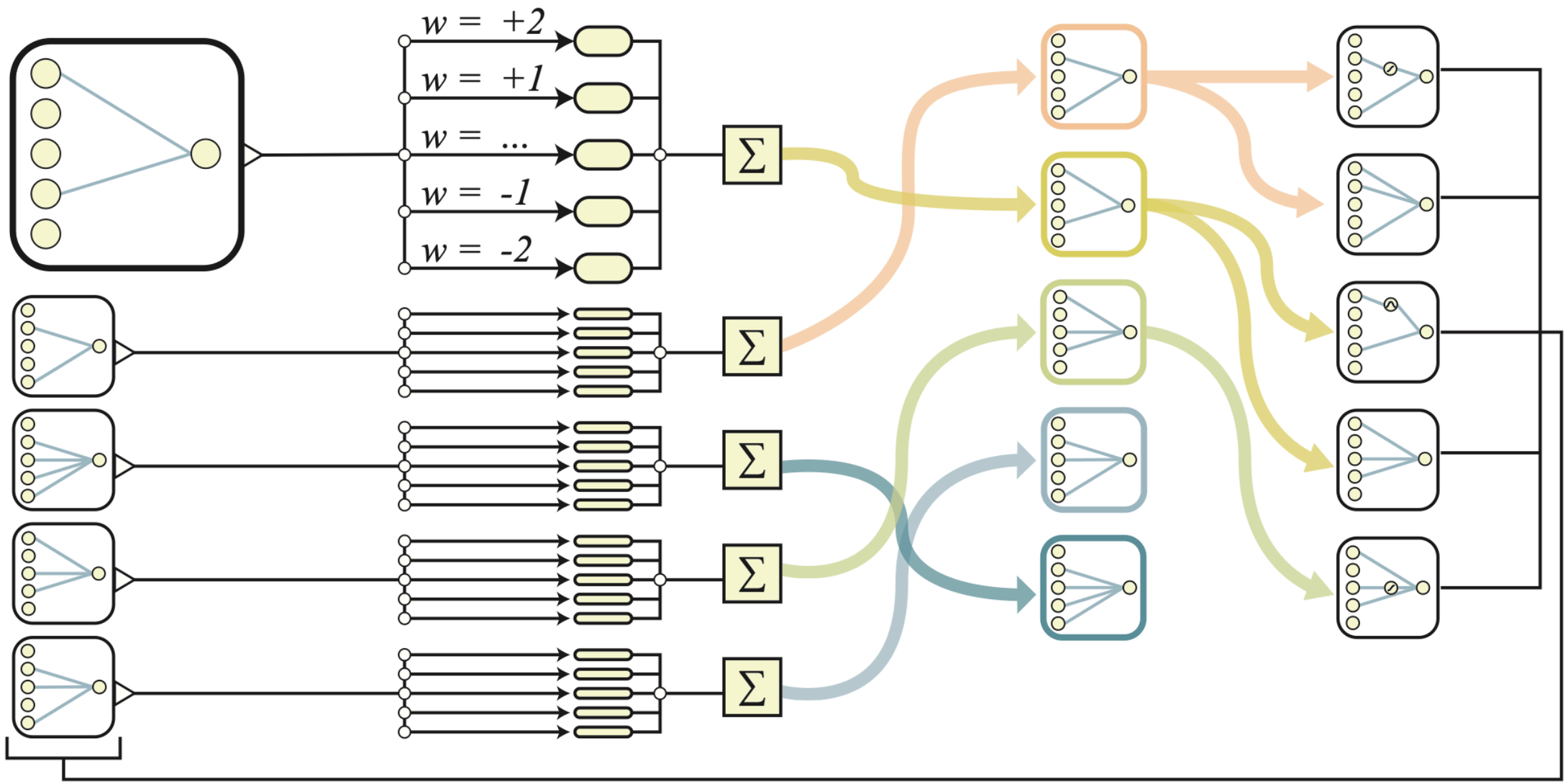
Method - Weight Agnostic Neural Network Search

1.) Initialize
Create population of minimal networks.

2.) Evaluate
Test with range of shared weight values.

3.) Rank
Rank by performance and complexity

4.) Vary
Create new population by varying best networks.

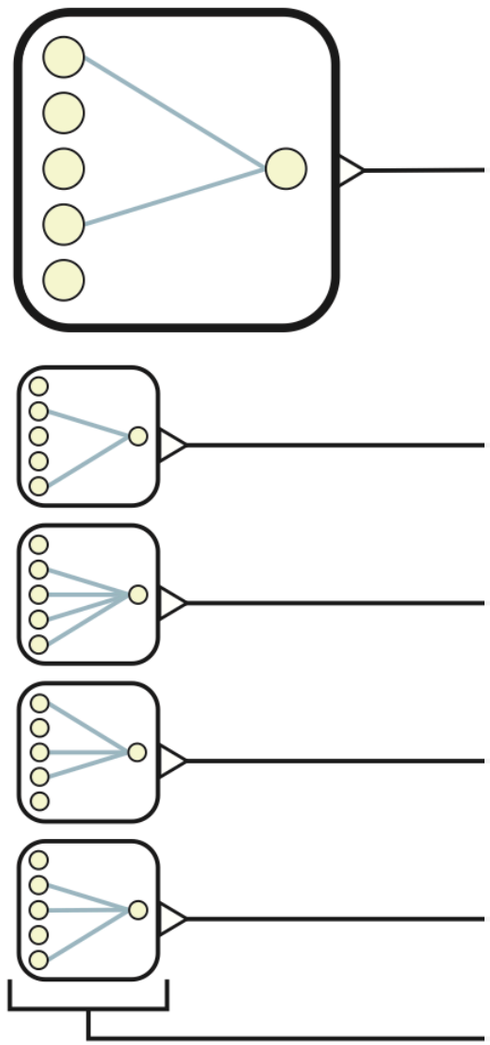


Method - Weight Agnostic Neural Network Search

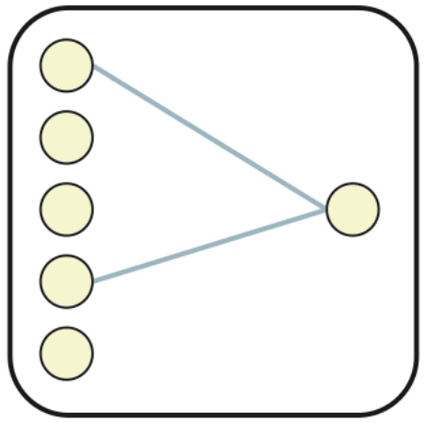
1.) Initialize
Create population of minimal networks.

Method - Weight Agnostic Neural Network Search

1.) Initialize
Create population of minimal networks.



Minimal Network

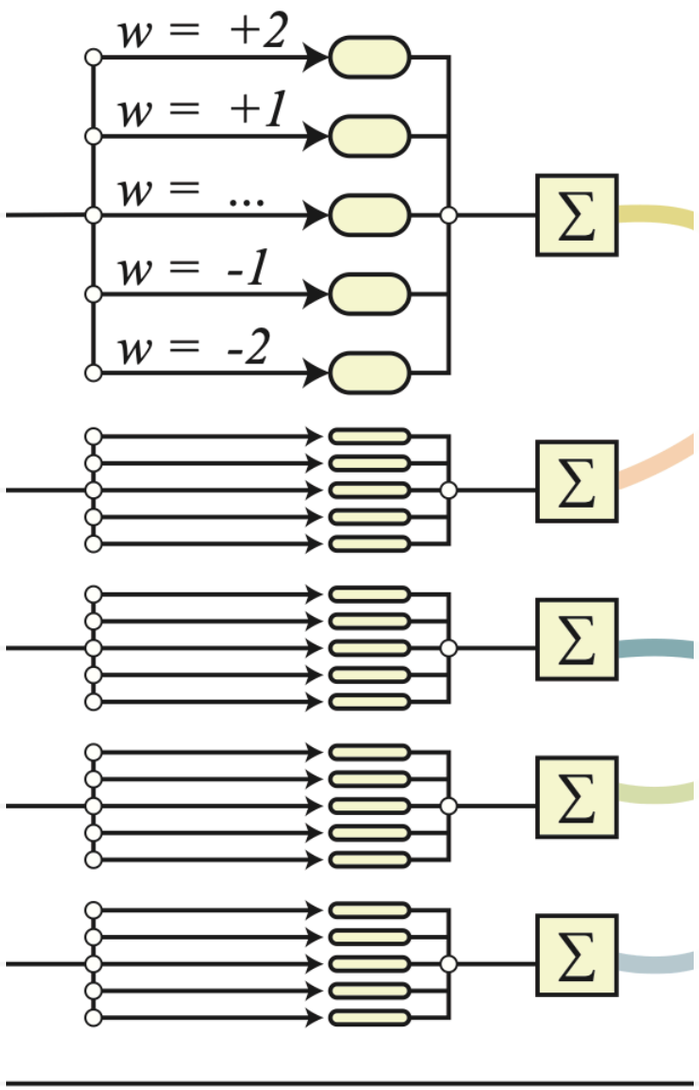


- Start with a population of minimal network topologies.
- **No hidden nodes.** Only a fraction of the possible connections between input and output.

Method - Weight Agnostic Neural Network Search

1.) Initialize
Create population of minimal networks.

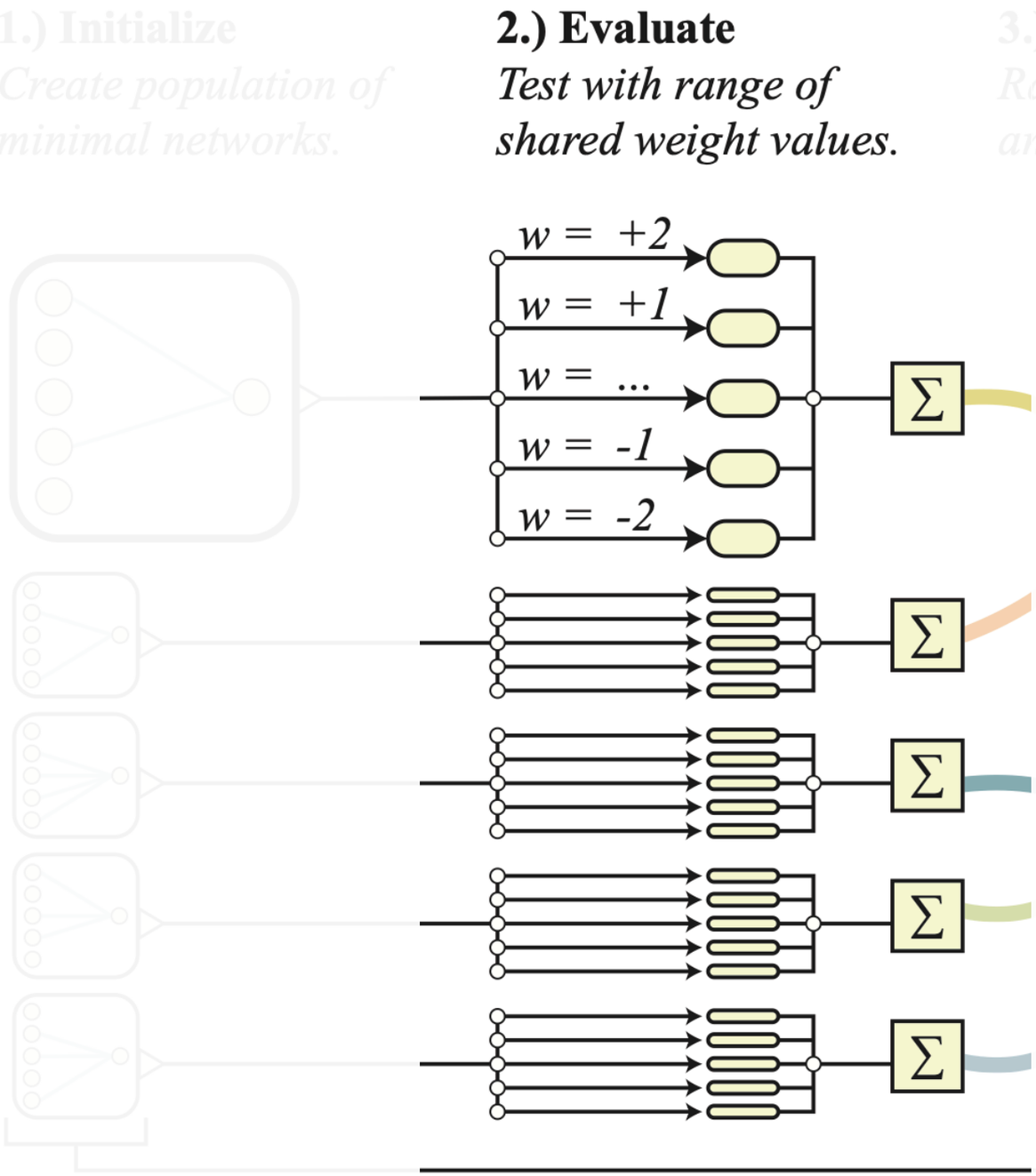
2.) Evaluate
Test with range of shared weight values.



3.) Rank
Rank by performance and complexity

4.) Vary
Create new population by varying best networks.

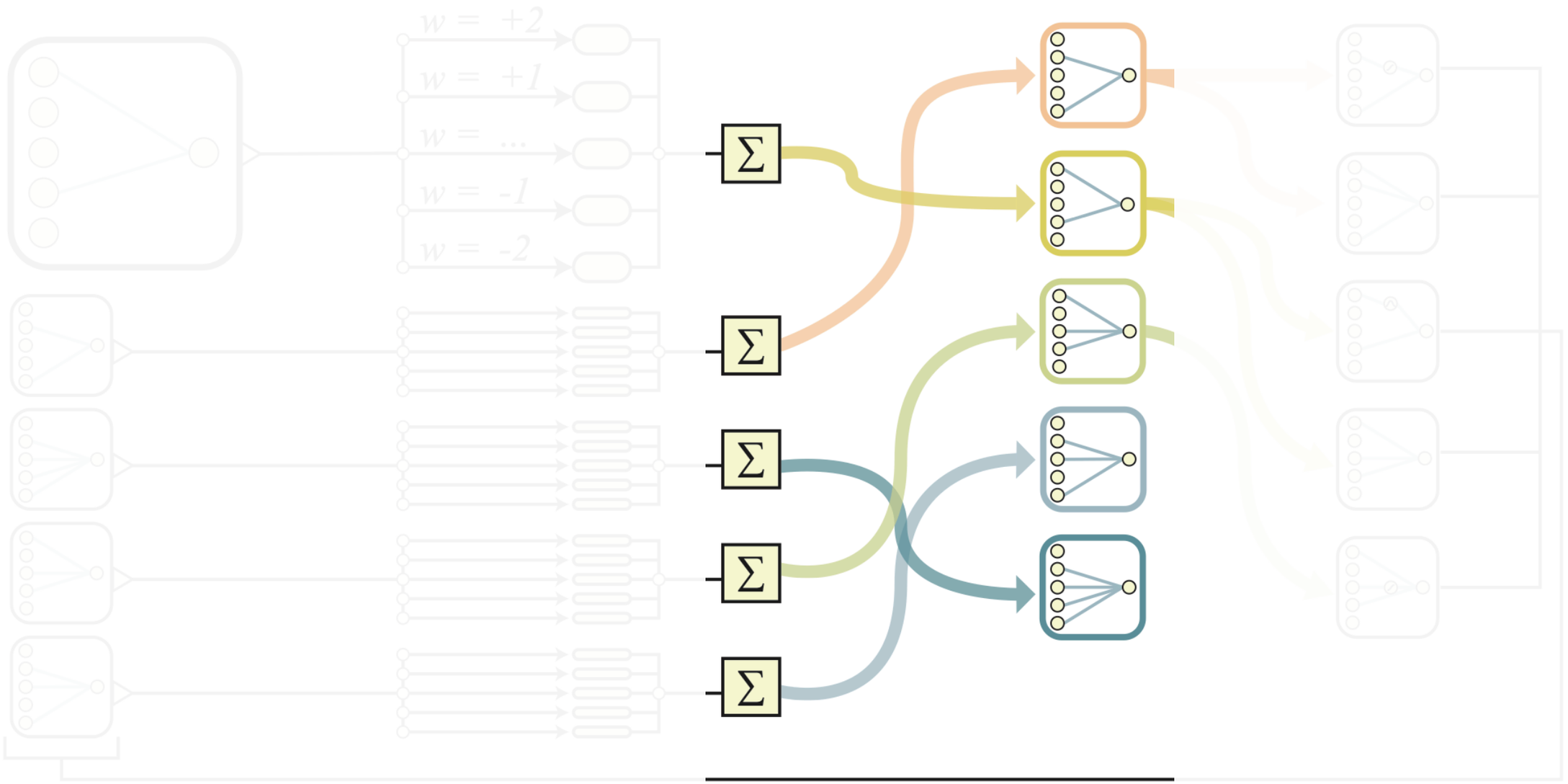
Method - Weight Agnostic Neural Network Search



- Each network is evaluated over **multiple rollouts.**
- Each rollout has different **shared weight value** assigned to **all connections.**
- Use **fixed** series of weight values (-2, -1, -0.5, +0.5, +1, +2). (assume U[-2,2] in test).
- Obtain **mean performance** and **max performance**.

Method - Weight Agnostic Neural Network Search

3.) Rank
Rank by performance and complexity

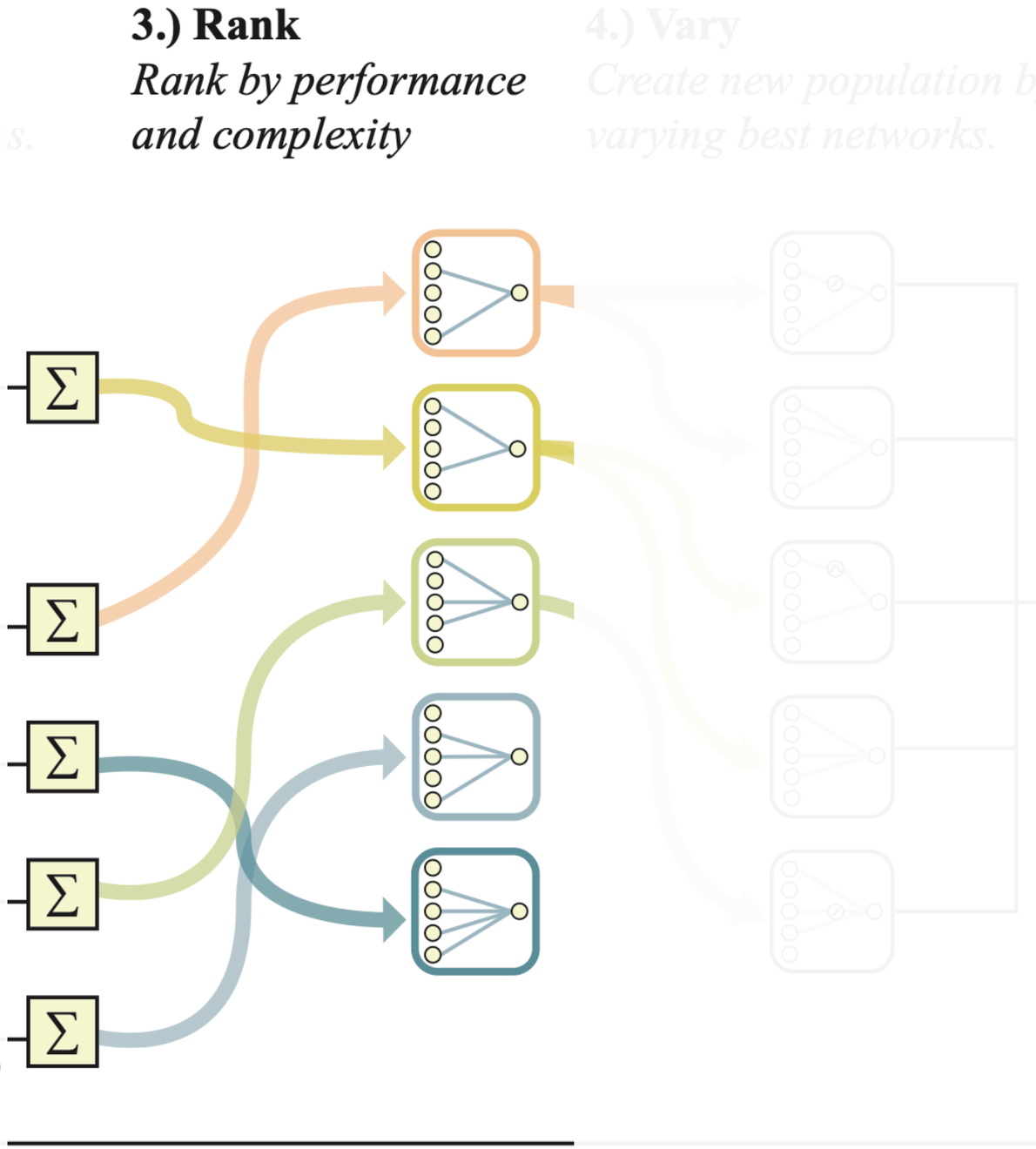


Method - Weight Agnostic Neural Network Search

- **Multi-objective** optimization:
  - Mean performance
  - Max performance
  - # Connections (complexity)

- No linear order: sometimes # Connections ↑ but mean/max performance ↓. **Sorting based on dominance relationship.**

- In 80% cases: ranked by (mean performance, # connections), in 20% cases: ranked by (mean performance, *max performance*)

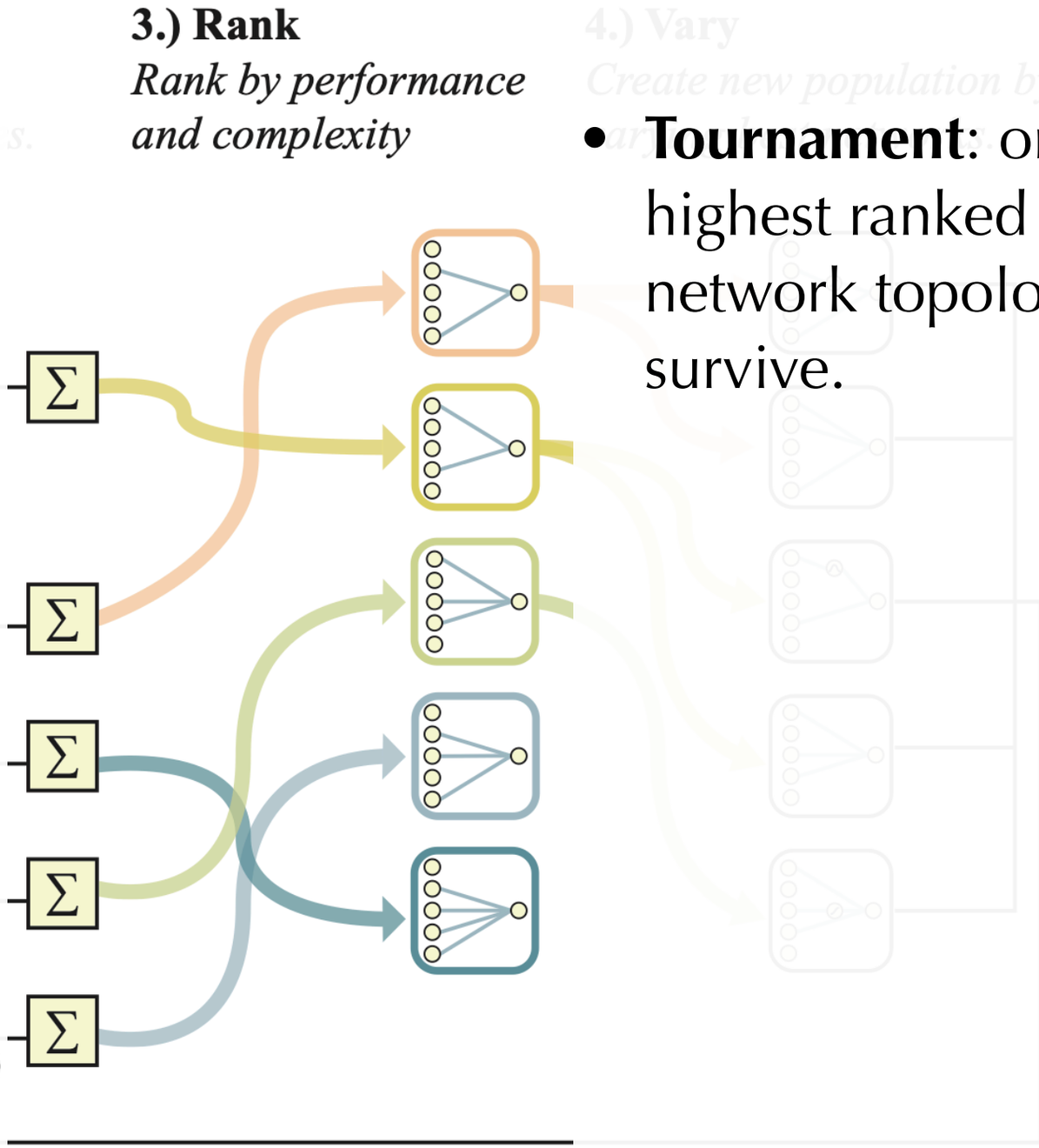


Method - Weight Agnostic Neural Network Search

- **Multi-objective** optimization:
  - Mean performance
  - Max performance
  - # Connections (complexity)

- No linear order: sometimes # Connections ↑ but mean/max performance ↓. **Sorting based on dominance relationship.**

- In 80% cases: ranked by (mean performance, # connections), in 20% cases: ranked by (mean performance, *max performance*)

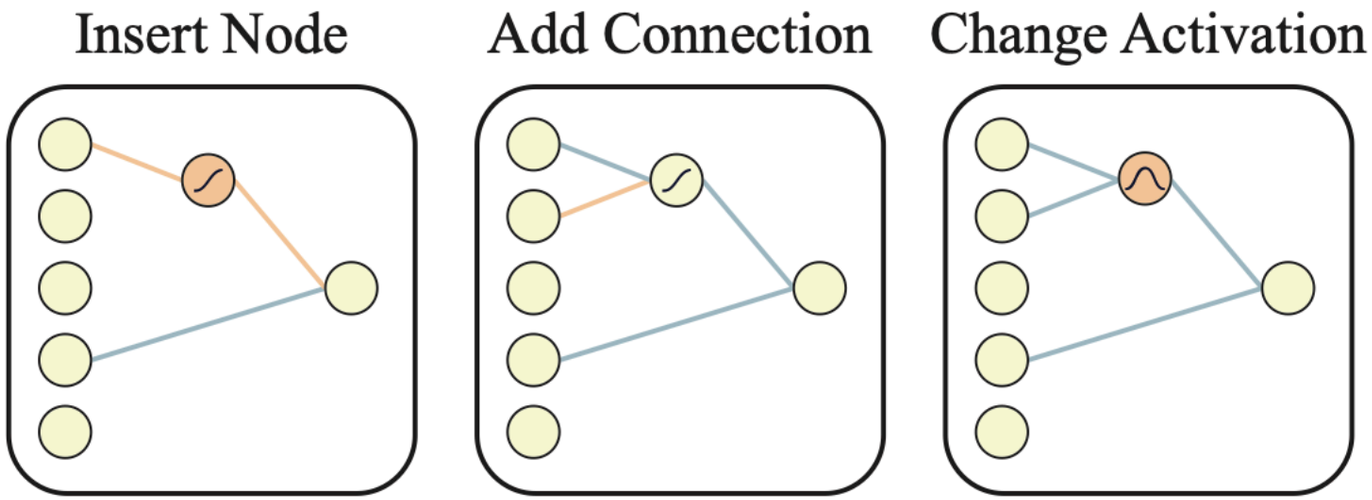


- **Tournament**: only the highest ranked network topologies survive.

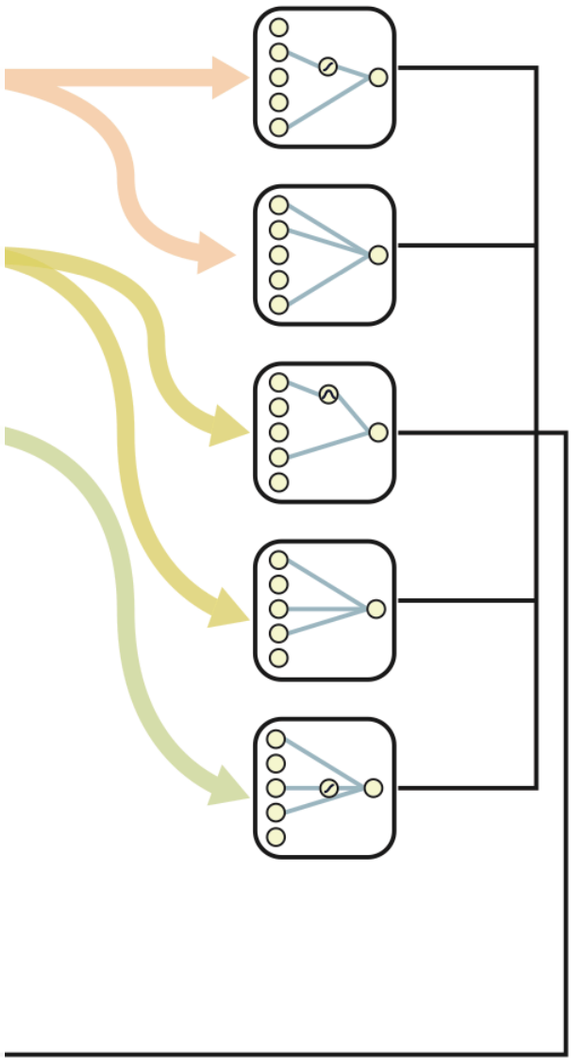
Method - Weight Agnostic Neural Network Search

4.) Vary
Create new population by varying best networks.

Method - Weight Agnostic Neural Network Search



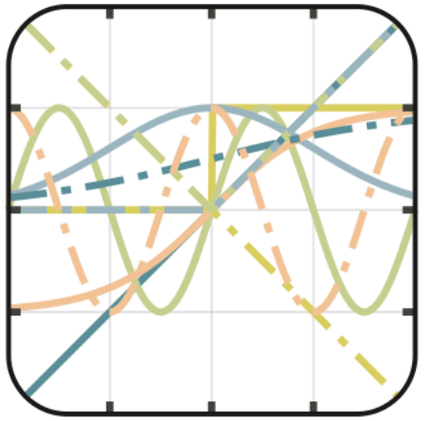
4.) Vary
Create new population by varying best networks.



- **Insert Node:** a new node is inserted by splitting an existing connection.
- **Add Connection:** a new connection is added by connecting two previously unconnected nodes.
- **Change Activation:** the activation function of a hidden node is reassigned.

Method - Weight Agnostic Neural Network Search

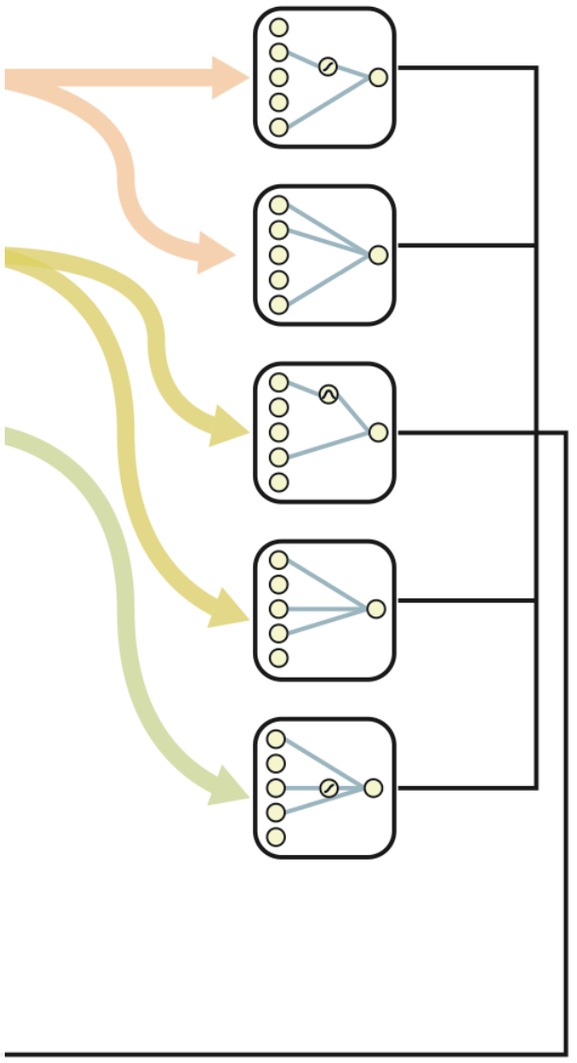
Node Activations



Possible activation functions contain:
linear, step, sin, cosine, Gaussian, tanh, sigmoid, inverse, absolute value, ReLU

negation…

4.) Vary
Create new population by varying best networks.



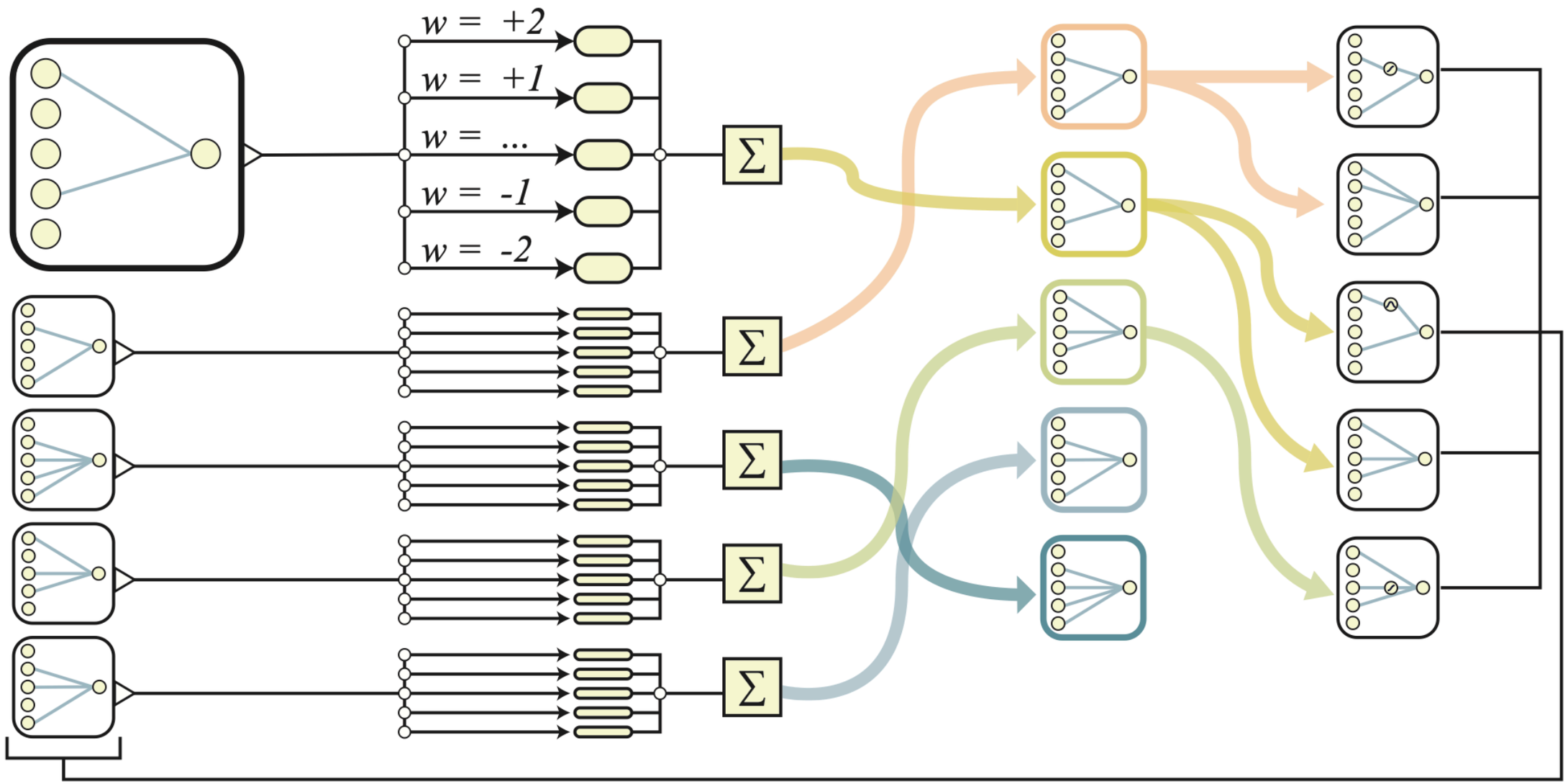
Method - Weight Agnostic Neural Network Search

1.) Initialize
Create population of minimal networks.

2.) Evaluate
Test with range of shared weight values.

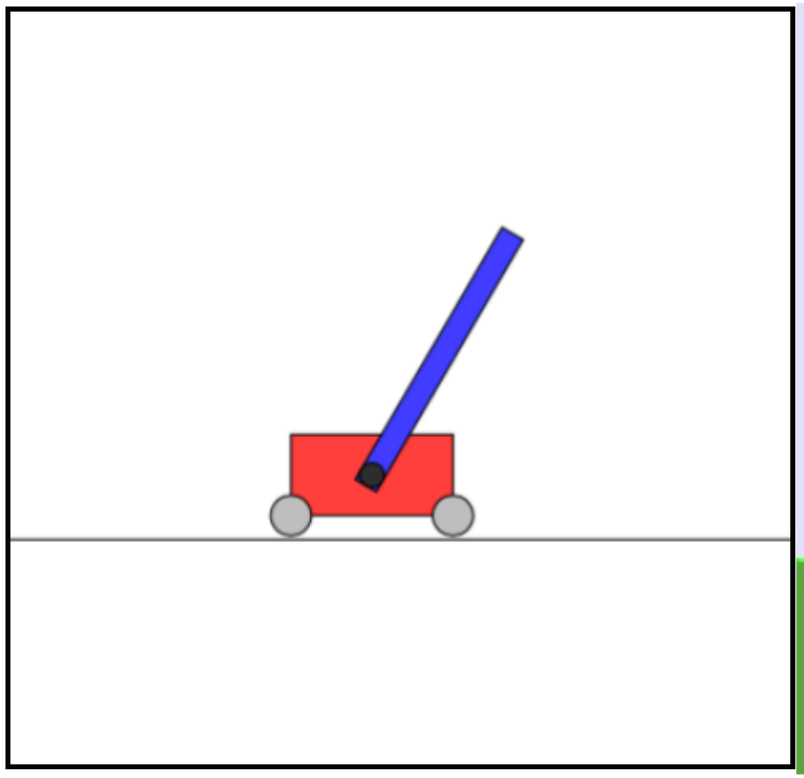
3.) Rank
Rank by performance and complexity

4.) Vary
Create new population by varying best networks.

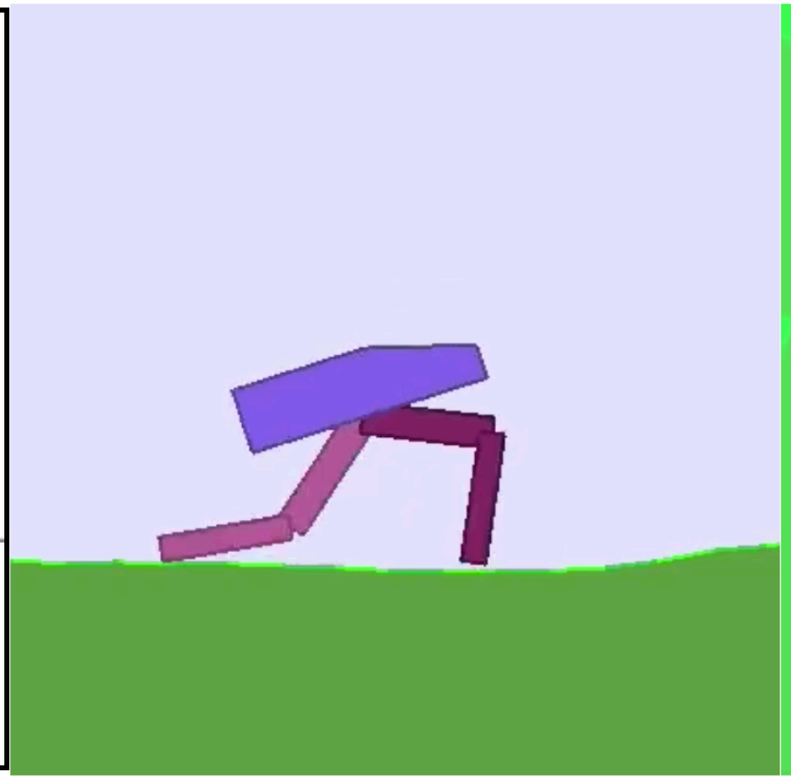


Experiment - Continuous Control

Three tasks:



CartPoleSwingUp



BipedalWalker-v2

CarRacing-v0

- Inspired by biological facts: lizard 🦎, snake 🐍 , …

Experiment - Continuous Control

Evaluation:

	W_{ij}~U(-2,2)	W~U(-2,2)	Opt.W	Opt.W_{ij}
Swing Up	Random Weights	Random Shared Weight	Tuned Shared Weight	Tuned Weights
WANN	**57 ± 121**	**515 ± 58**	**723 ± 16**	**932 ± 6**
Fixed Topology	21 ± 43	7 ± 2	8 ± 1	918 ± 7

Biped	Random Weights	Random Shared Weight	Tuned Shared Weight	Tuned Weights
WANN	**-46 ± 54**	**51 ± 108**	**261 ± 58**	332 ± 1
Fixed Topology	-129 ± 28	-107 ± 12	-35 ± 23	**347 ± 1** [38]

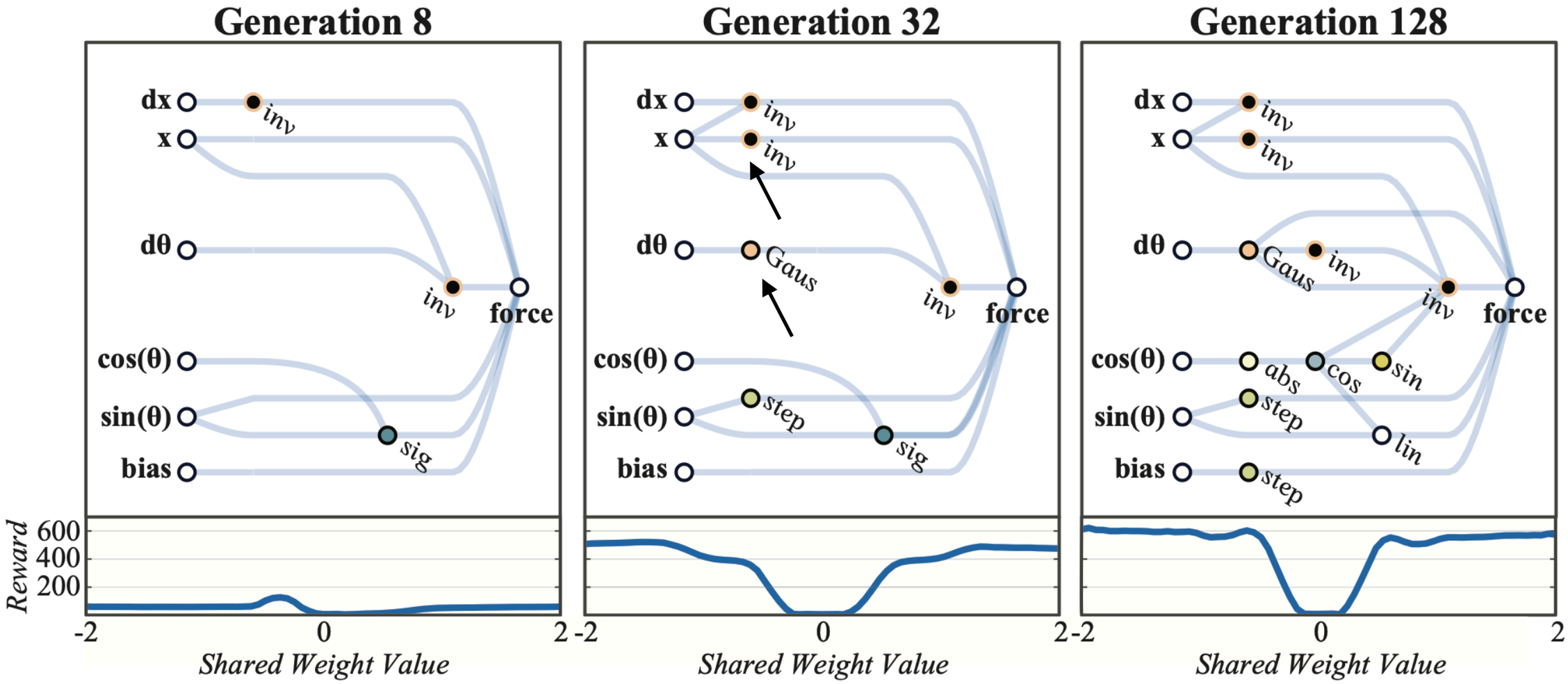
CarRacing	Random Weights	Random Shared Weight	Tuned Shared Weight	Tuned Weights
WANN	**-69 ± 31**	**375 ± 177**	**608 ± 161**	893 ± 74
Fixed Topology	-82 ± 13	-85 ± 27	-37 ± 36	**906 ± 21** [39]

Performance of Randomly Sampled and Trained Weights for Continuous Control Tasks

The mean performance (over 100 trials) of the **best weight agnostic network architectures** found are compared with **standard feed forward network policies** commonly used in previous work (SOTA baselines for Biped and for CarRacing).

Experiment - Continuous Control

Network Evolution:

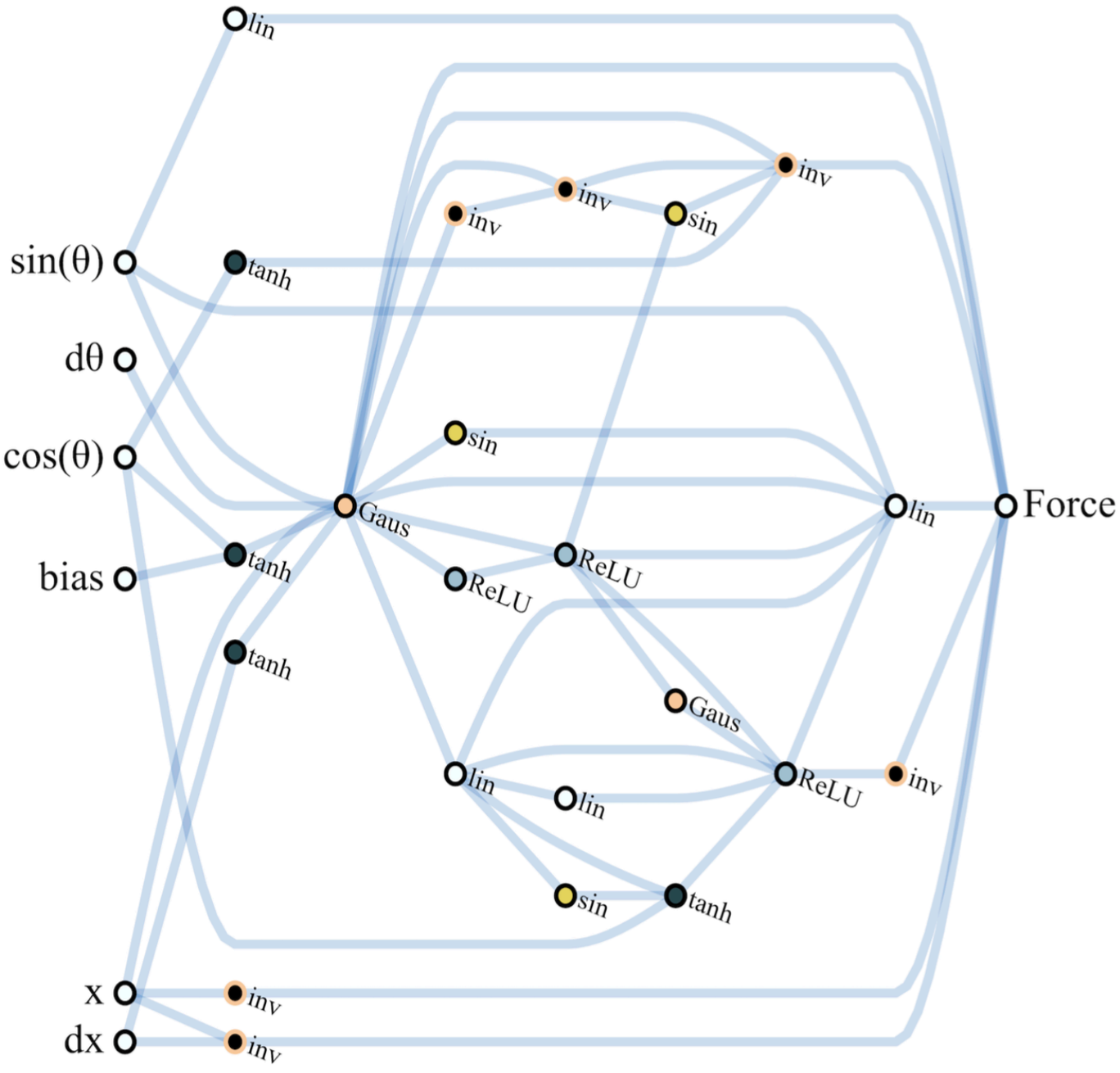


Development of Weight Agnostic topologies over time

G8: An early network which performs poorly with nearly all weights.
G32: Relationships between the position of the cart and velocity of the pole are established.
G128: Complexity is added to refine the balancing behavior of the elevated pole.

Experiment - Continuous Control

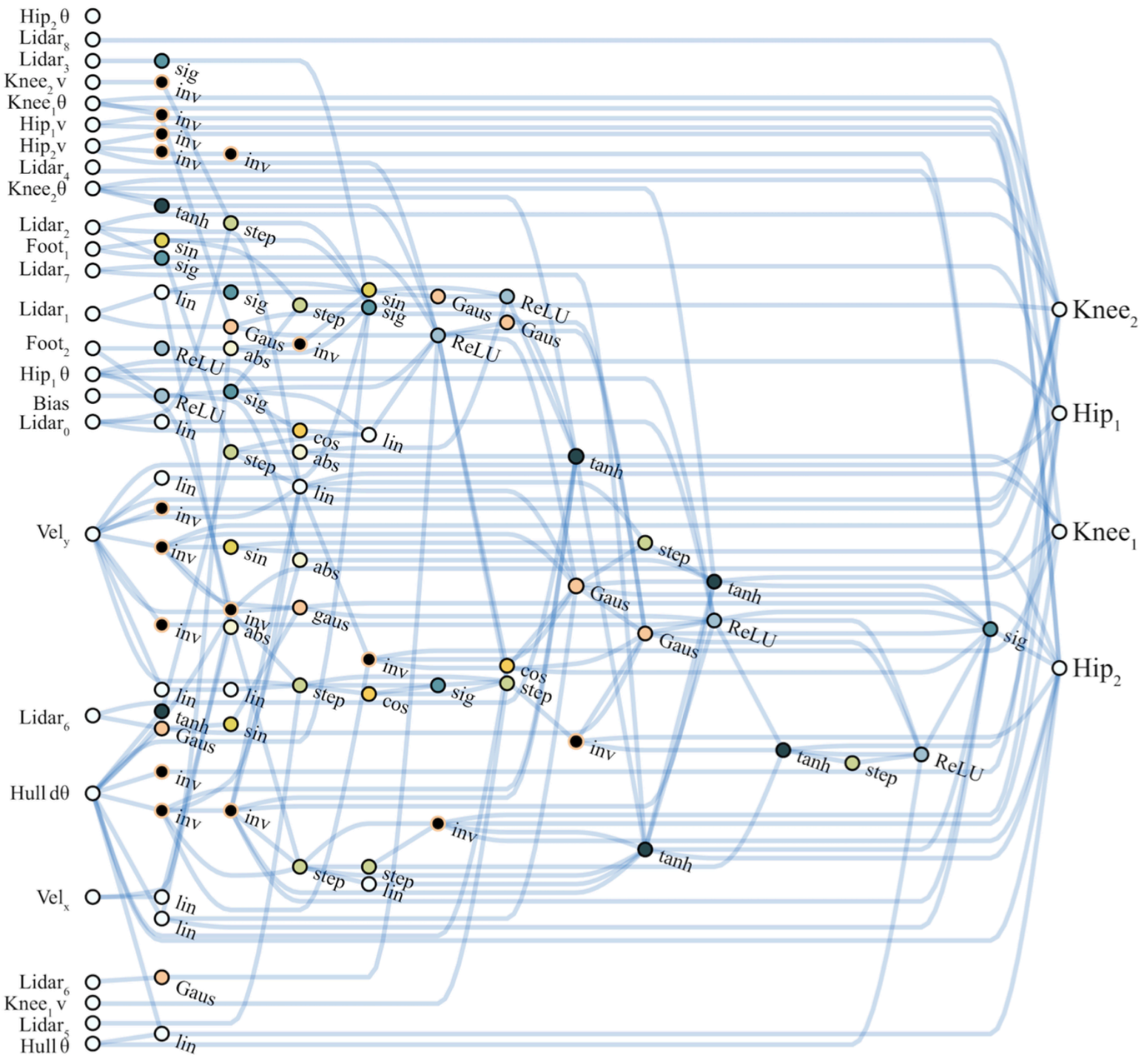
Champion networks:



CartpoleSwingUp champion network

Experiment - Continuous Control

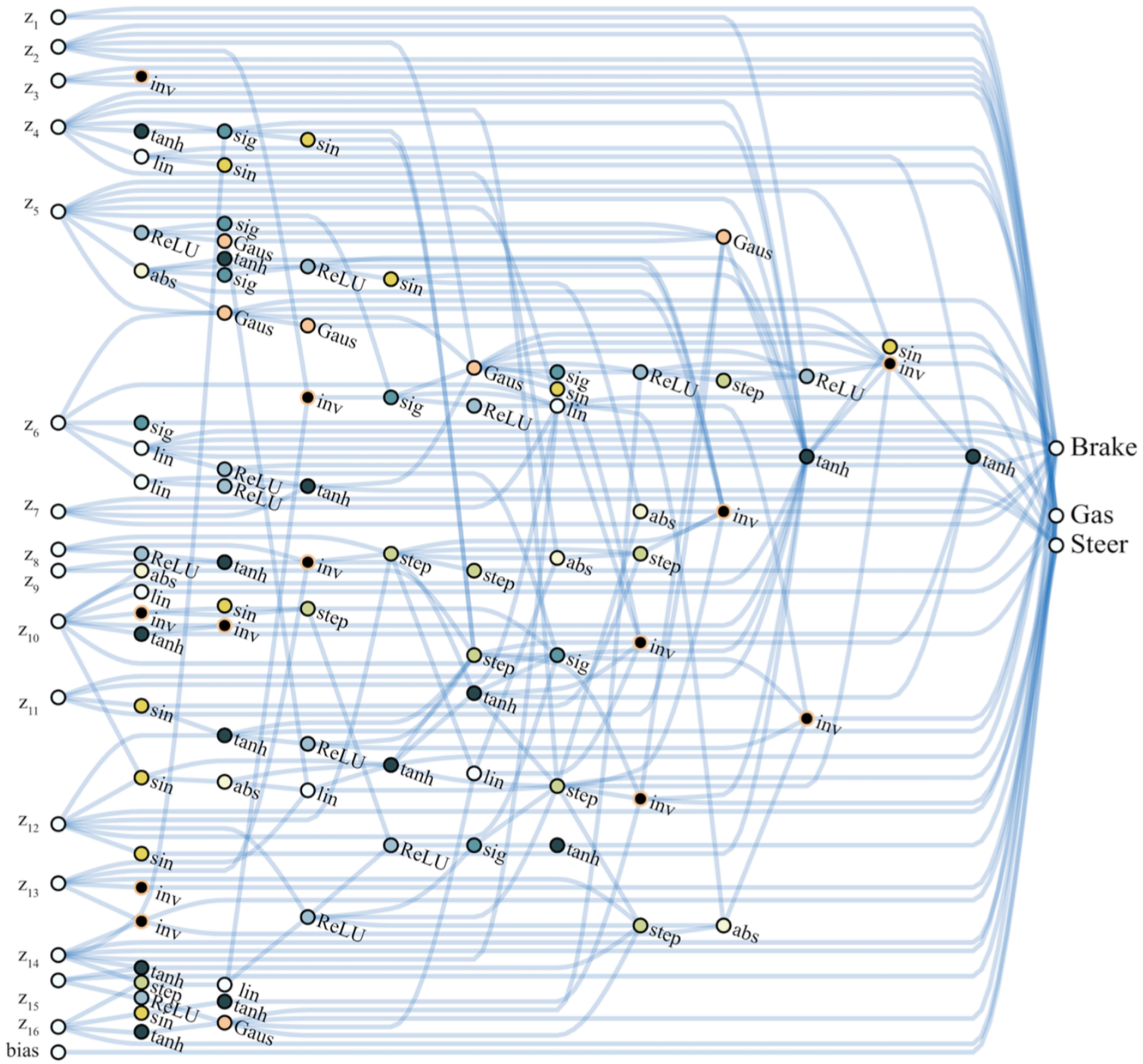
Champion networks:



BipedalWalker champion network

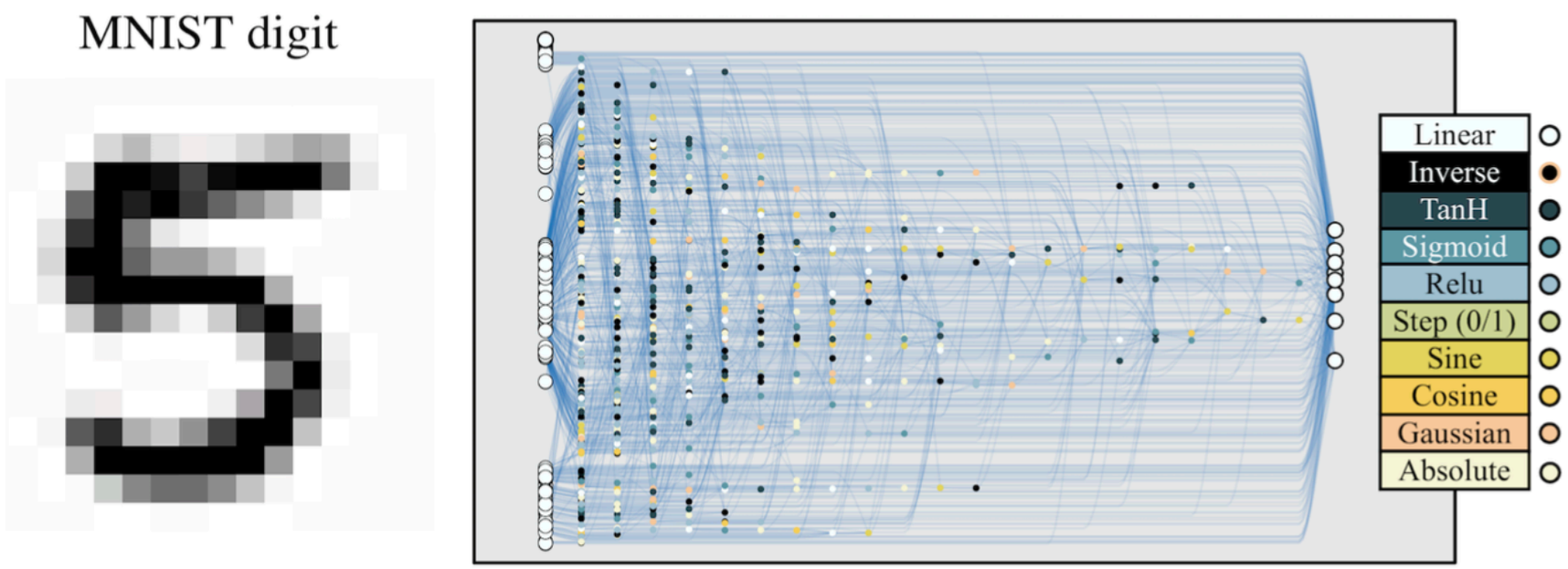
Experiment - Continuous Control

Champion networks:



Champion network for *CarRacing-v0*

Experiment - Classification



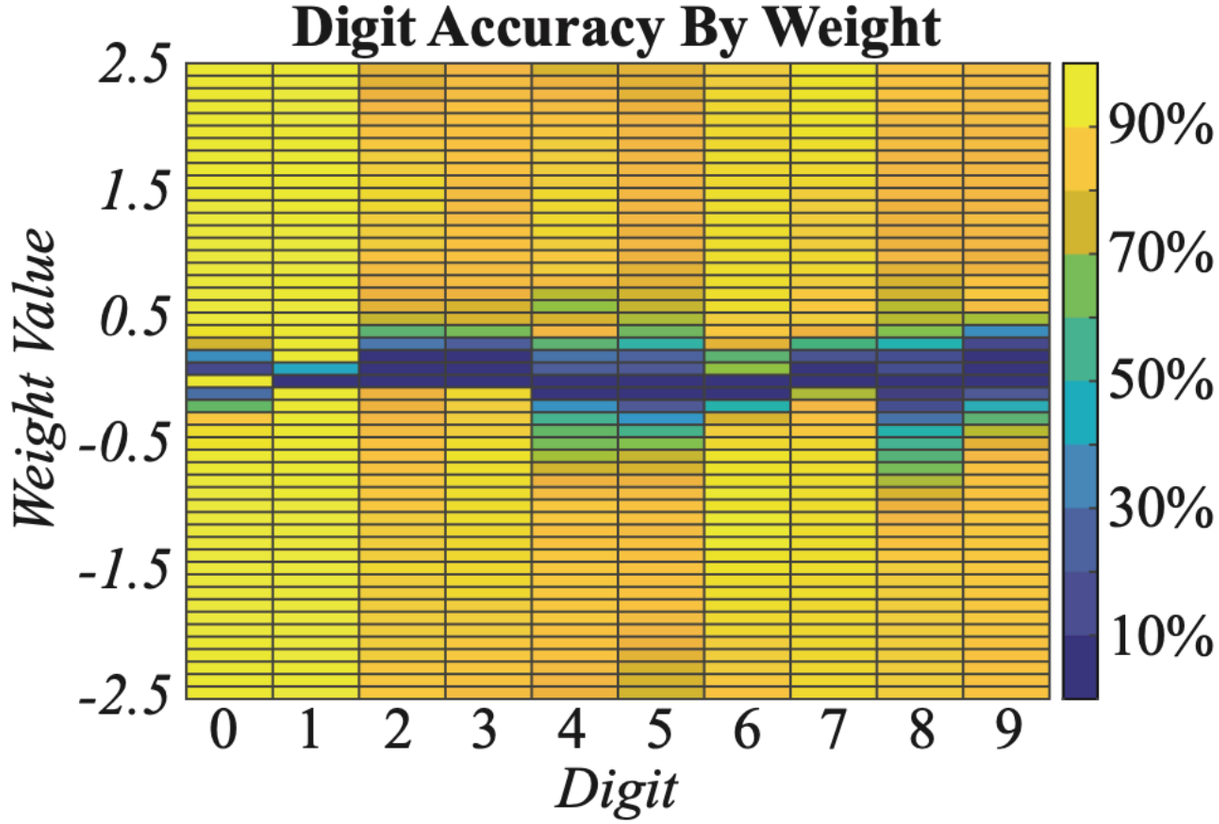
- The design of architectures is a focus for classification tasks. **"human-led architecture search"** on MNIST.

- High-dimensional inputs.

Experiment - Classification

WANN	Test Accuracy
Random Weight	*82.0% ± 18.7%*
Ensemble Weights	*91.6%*
Tuned Weight	*91.9%*
Trained Weights	*94.2%*

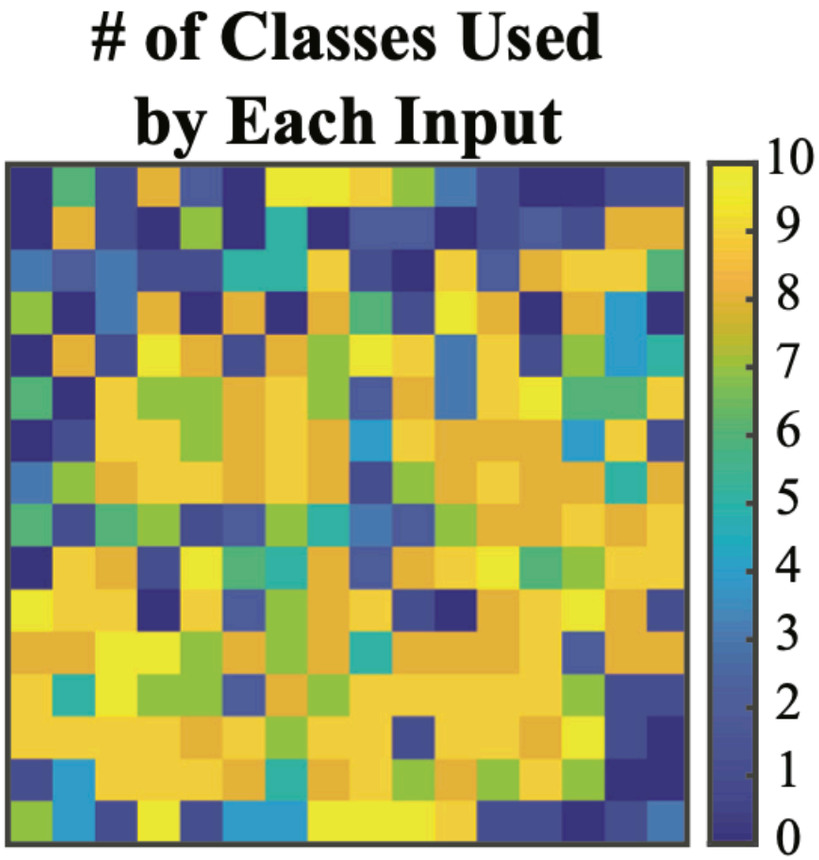
ANN	Test Accuracy
Linear Regression	*91.6%* [62]
Two-Layer CNN	*99.3%* [15]



Classification Accuracy on MNIST

WANNs instantiated with multiple weight values acting as an ensemble perform far better than when weights are sampled at random, and as well as a linear classifier with thousands of weights. **No single weight value has better accuracy on all digits**. The ensemble classifies samples according to the category which received the **most votes**.

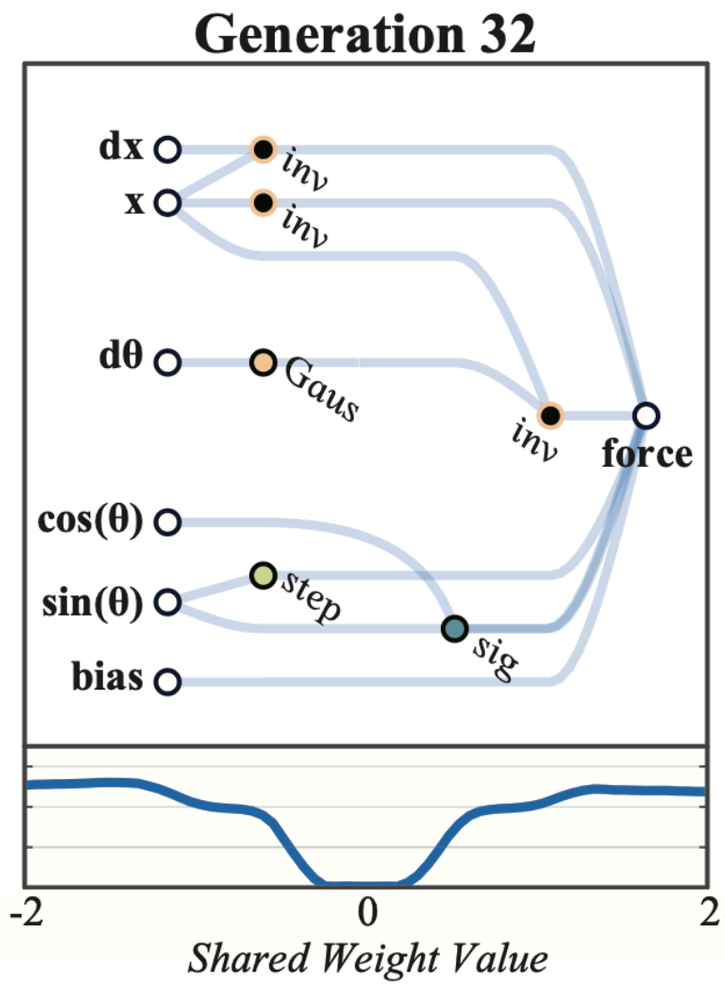
Experiment - Classification



Receptive Field:

Not all neurons and connections are used to predict each digit. Starting from the output connection for a particular digit, we can map out the part of the network used to classify that digit. We can also see which parts of the inputs are used for classification.

Discussion



Sample shared weights from a zero-mean uniform distribution?

? **?**

Discussion

- **Fine-tuning shared weights** is useful **in few-shot learning.**
- **Convolutional layers** are unbeatable?
- **Baldwin effect:** not the blank slates.
- **Language acquisition**: Chomsky's P&P theory?
- ...