Automatically Searching the Optimal Neural Network Structures

Spring 2018

The University of Iowa

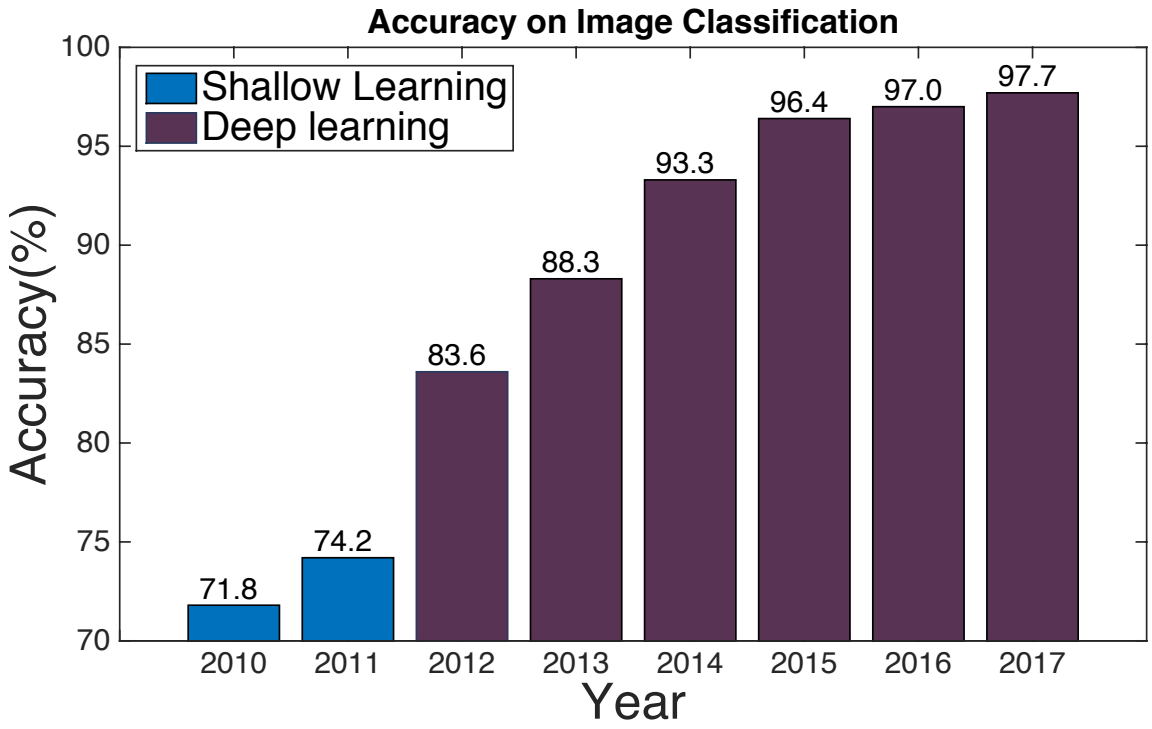
Zhe Li

Content

- **Motivation**
- Genetic Approach
- Reinforcement Learning Approach
- Experimental Results
- Summary

Motivation

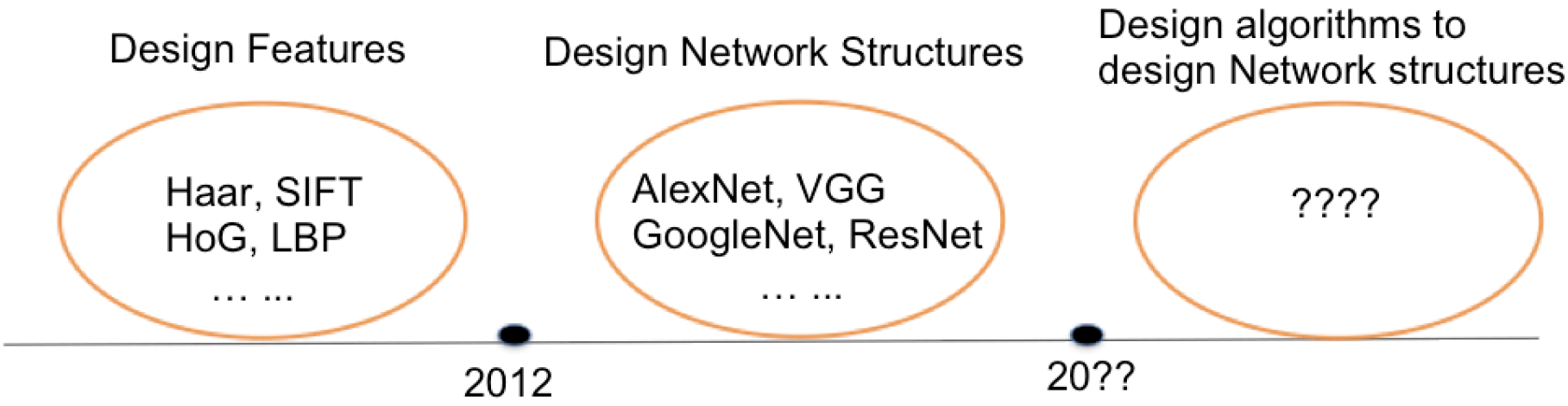
- Example: Image classification



Accuracy: Test on ImageNet Benchmark Dataset (about 1.2 million images, 1000 classes)

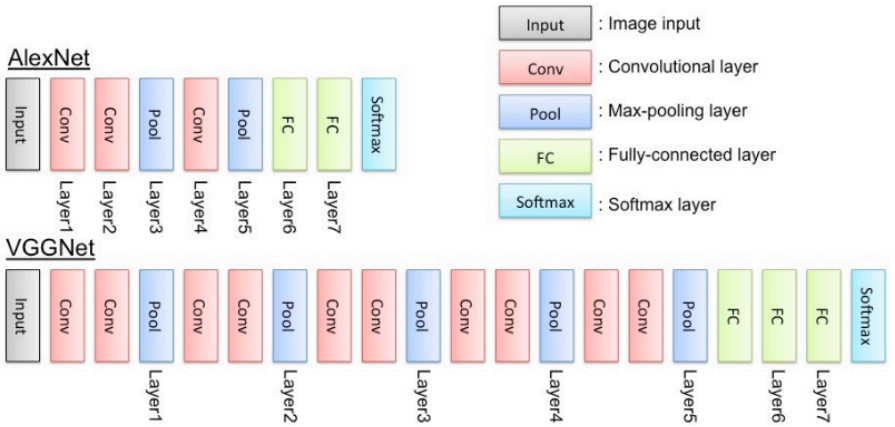
Motivation

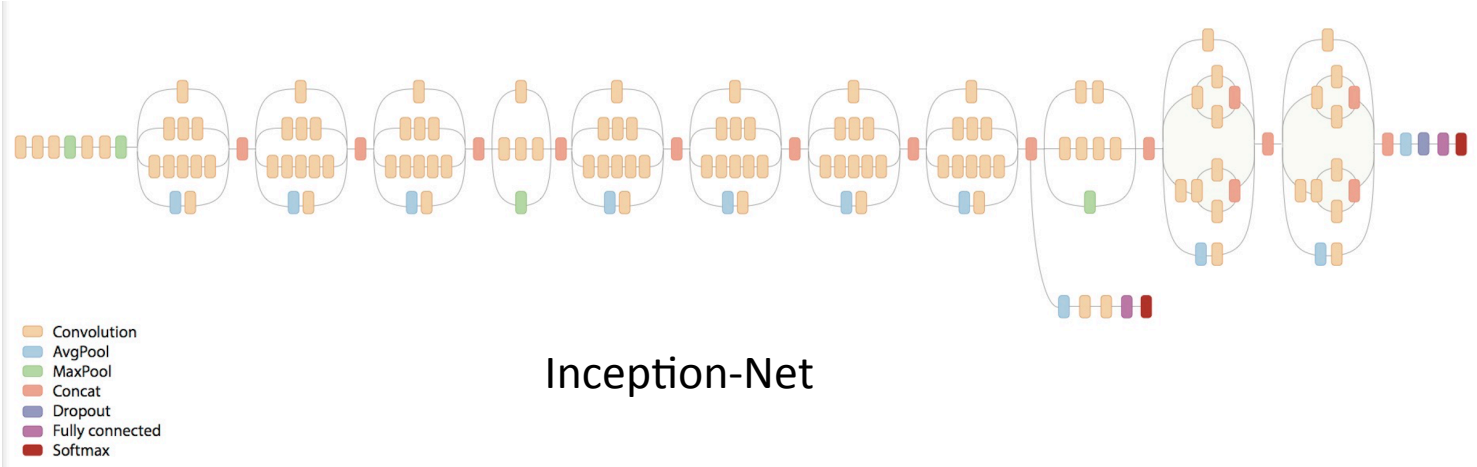
- Go through the history of the performance improvement for computer vision tasks



Motivation

- Alex-Net (2012)
- VGG-Net (2013)
- Inception-Net (2014)
- Res-Net (2015)
- Dense-Net (2016)





Inception-Net

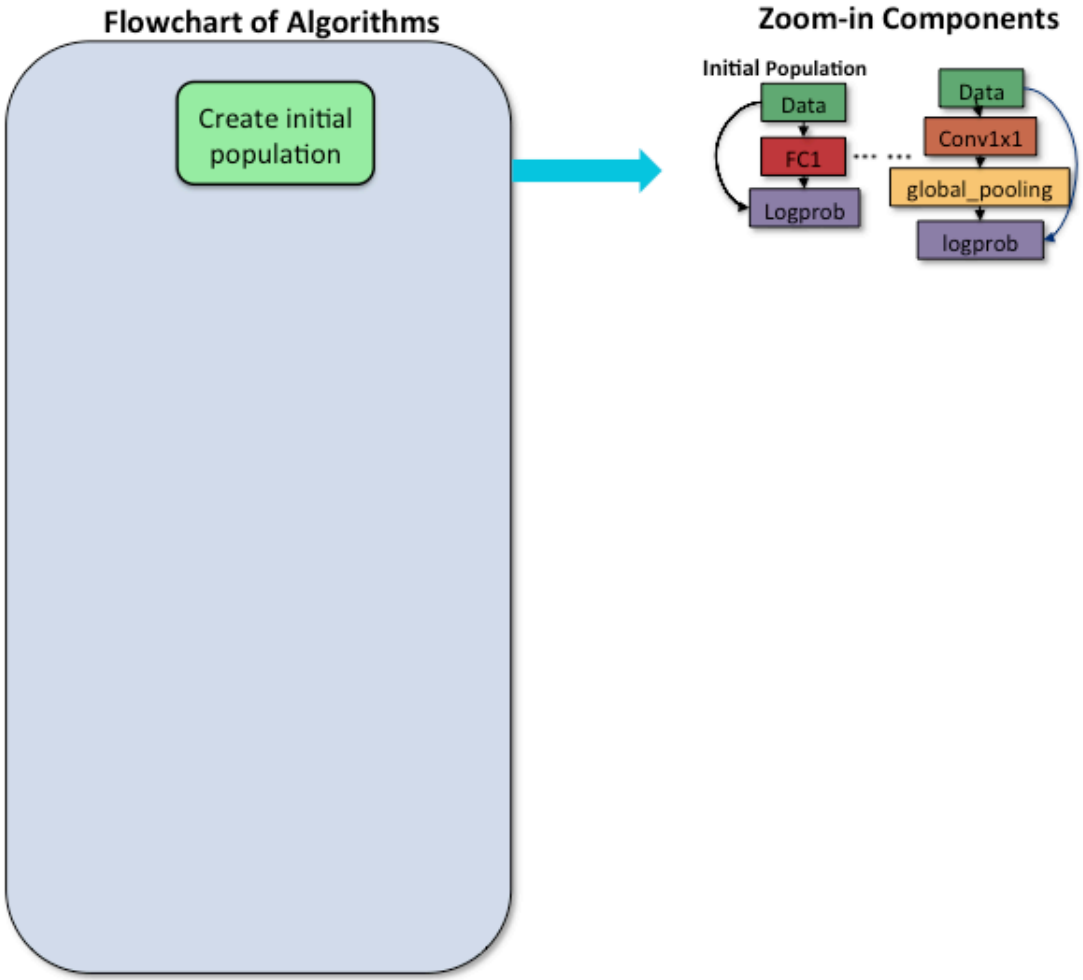
Motivation

- Can we develop an efficient algorithm to design/ search the optimal neural network structures?
  - Genetic Approach
  - Reinforcement Learning Approach

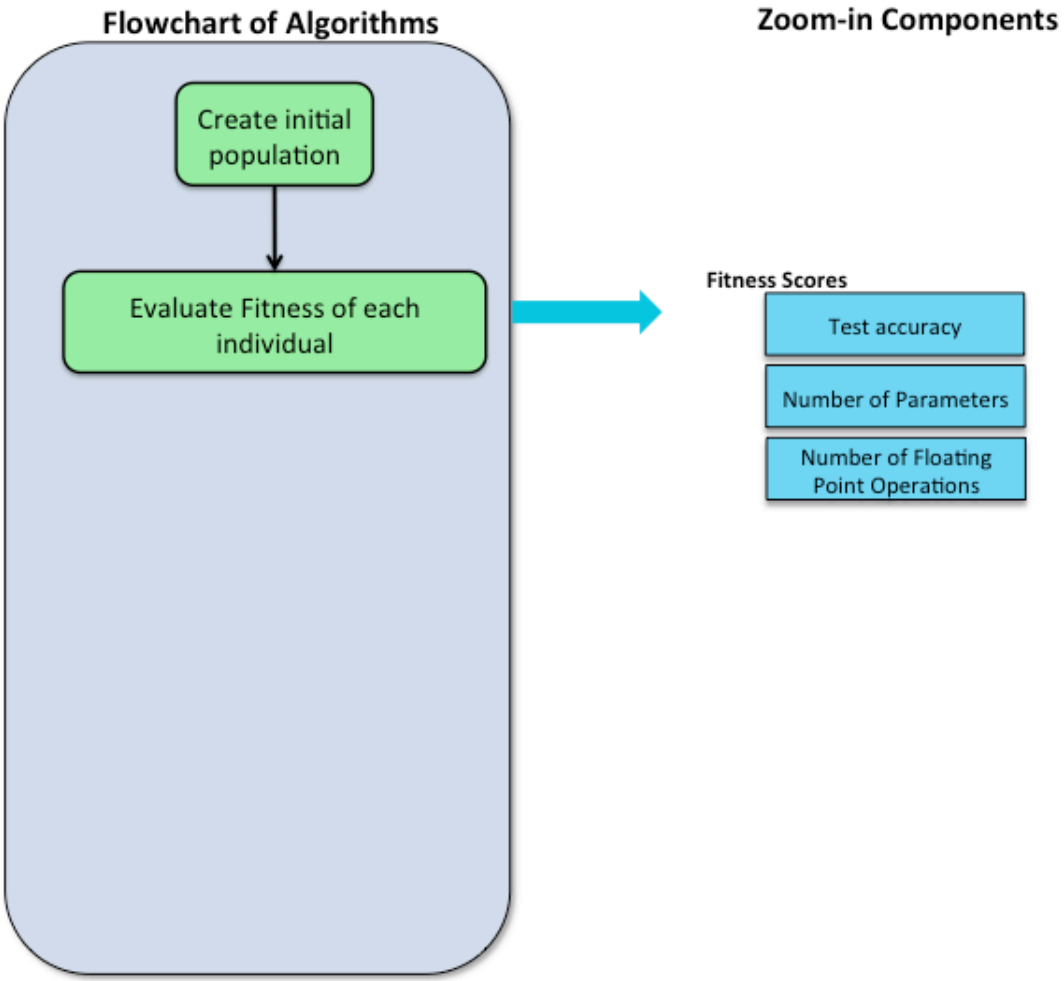
Content

- Motivation
- **Genetic Approach**
- Reinforcement Learning Approach
- Experimental Results
- Summary

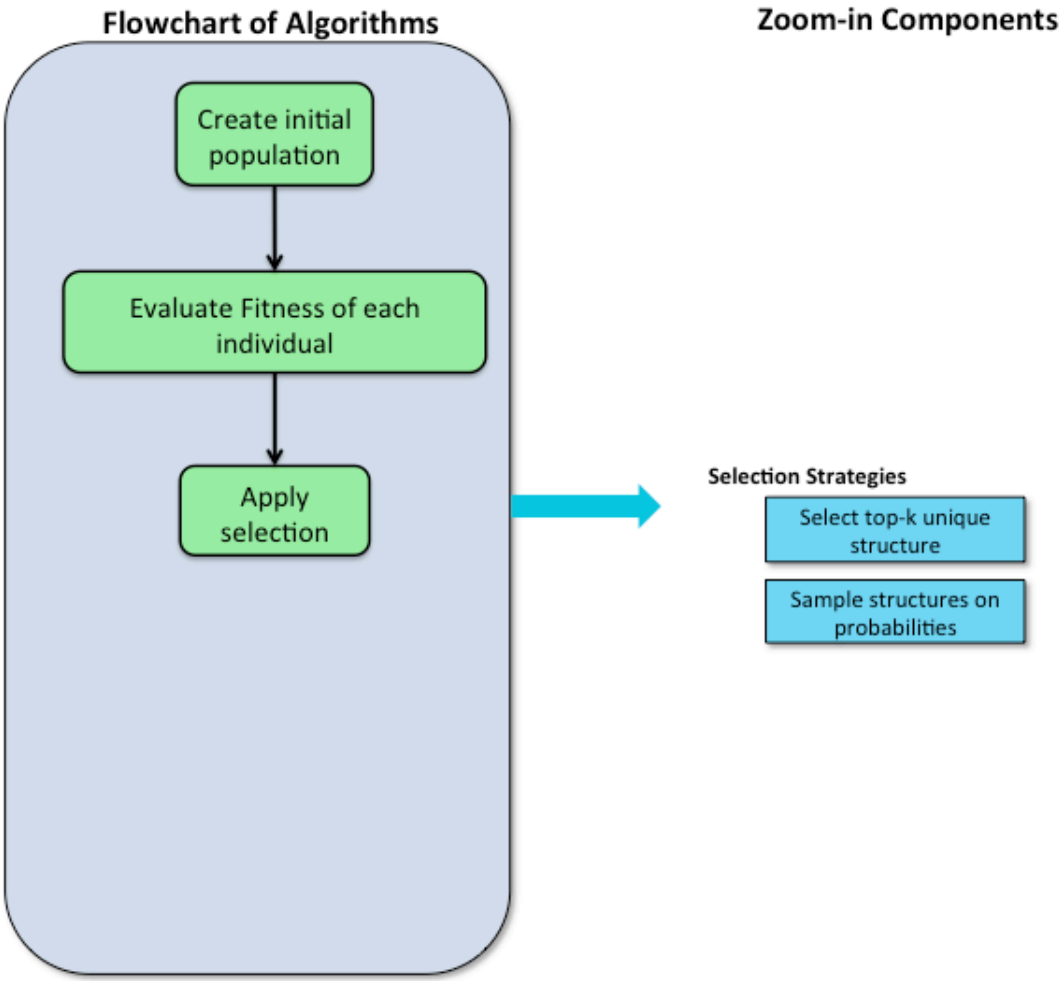
Genetic Approach



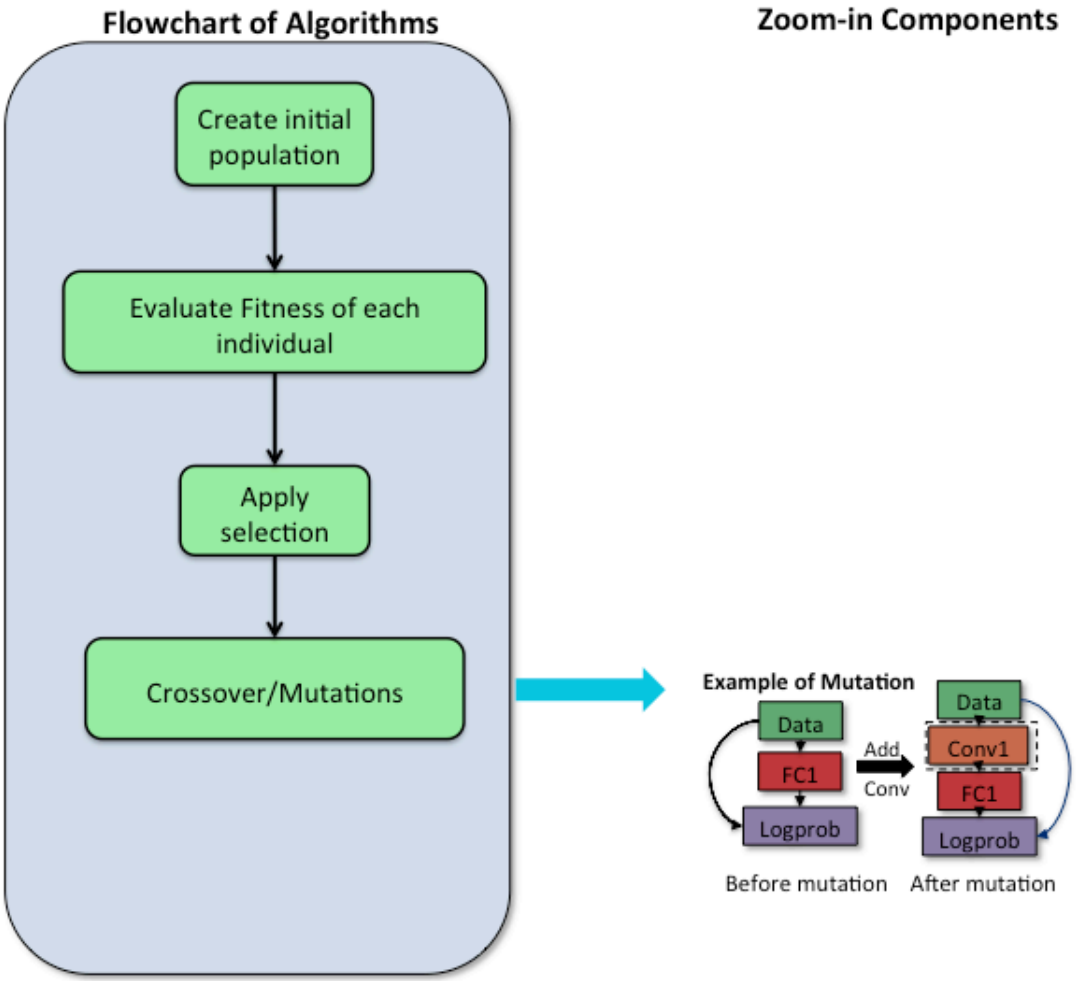
Genetic Approach



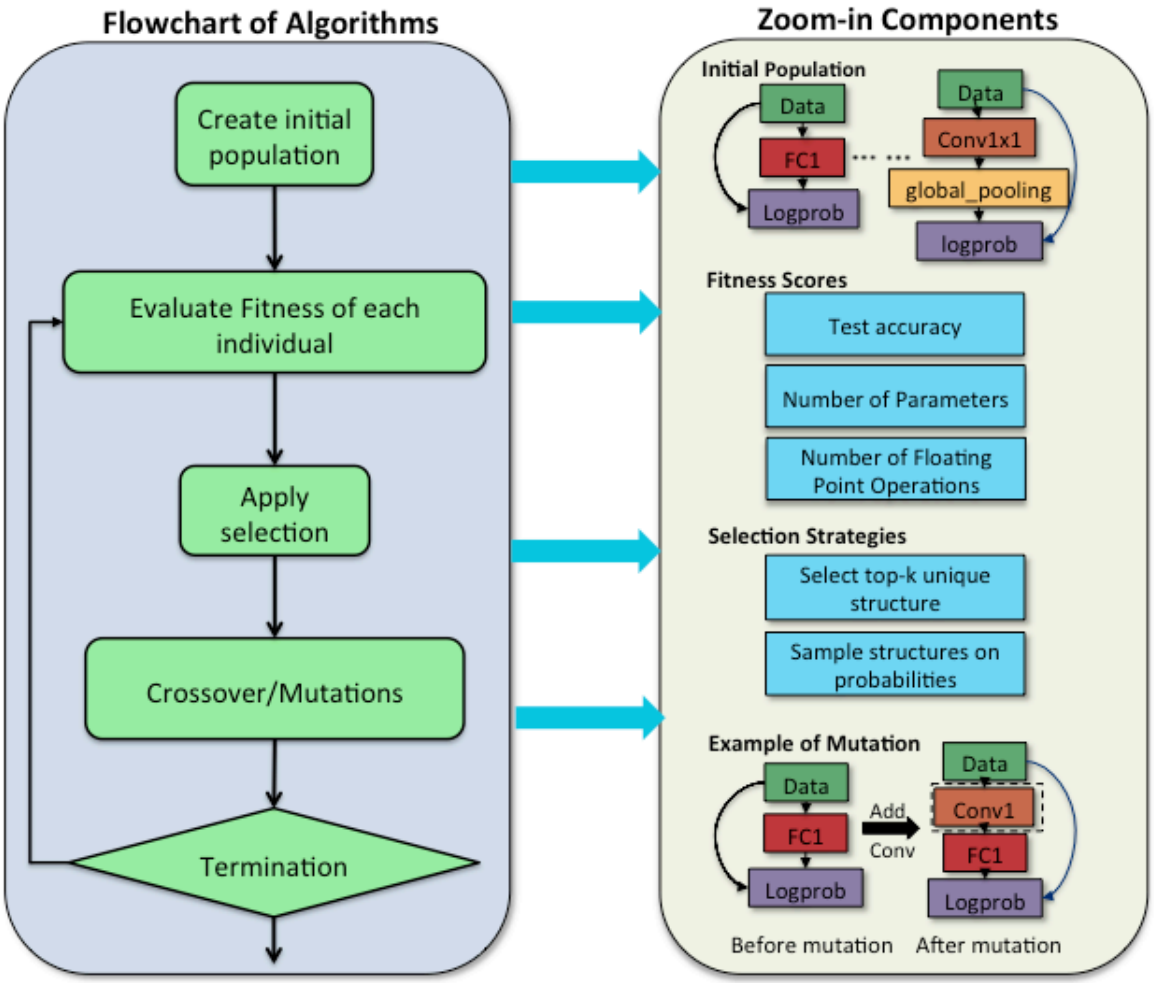
Genetic Approach



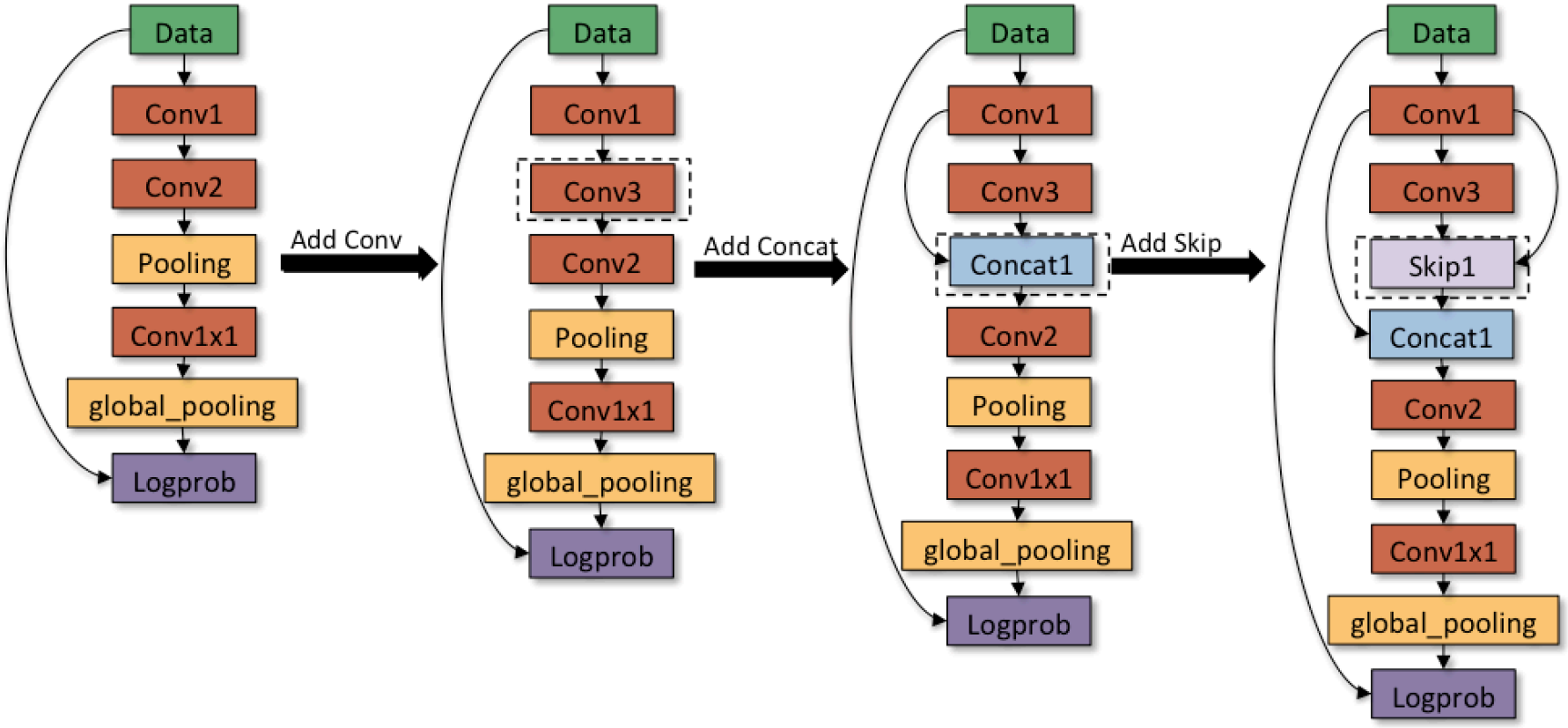
Genetic Approach



Genetic Approach



Example of Mutation Operations



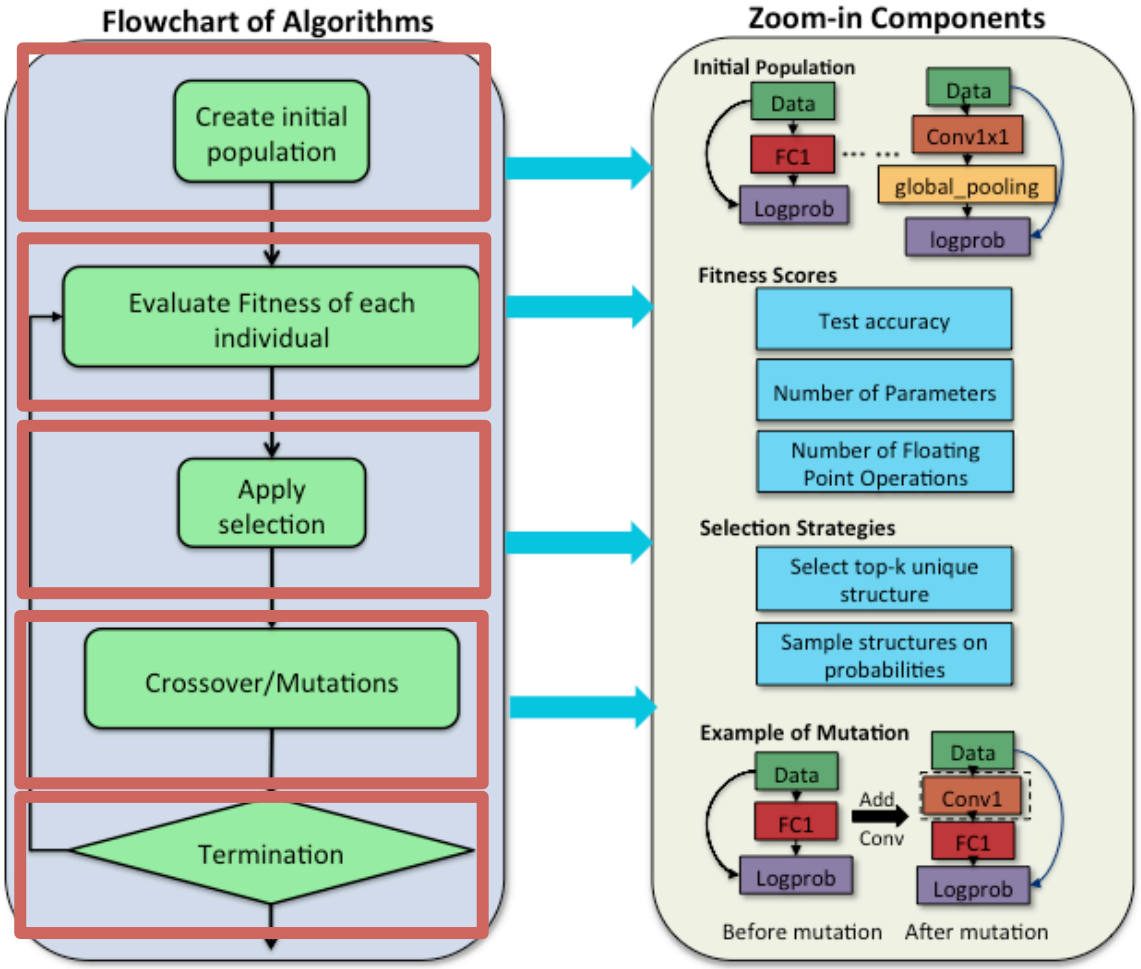
Challenges

- **Implementation:** Build this automatically searching framework

- **Computation cost:** training one neural network is computationally expensive, and we need to train a lot of neural networks by using genetic approach.

Reducing computational cost

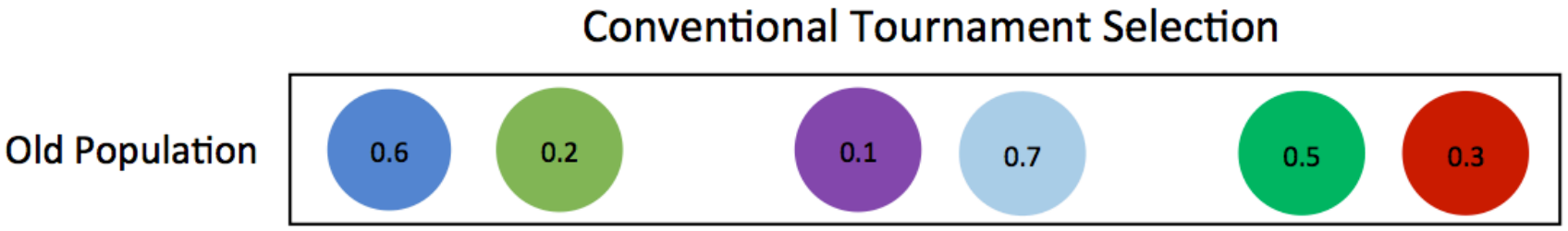
Which components are computationally expansive?



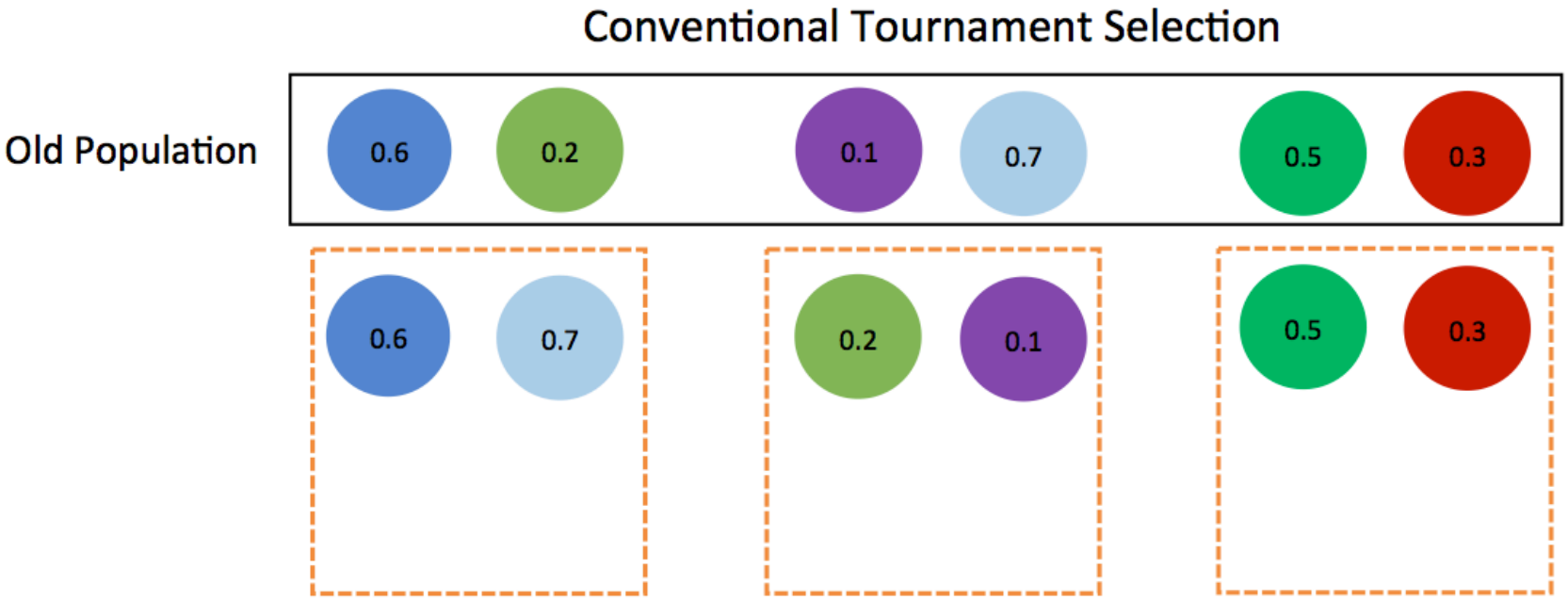
Selection Strategy

- Selection Strategy itself is not computationally expensive, but the consequence of selection has large influence on computational cost in future.
  - Random sampling
  - Random sampling based on fitness score
  - Tournament selection strategy
  - Aggressive selection strategy (Proposed)

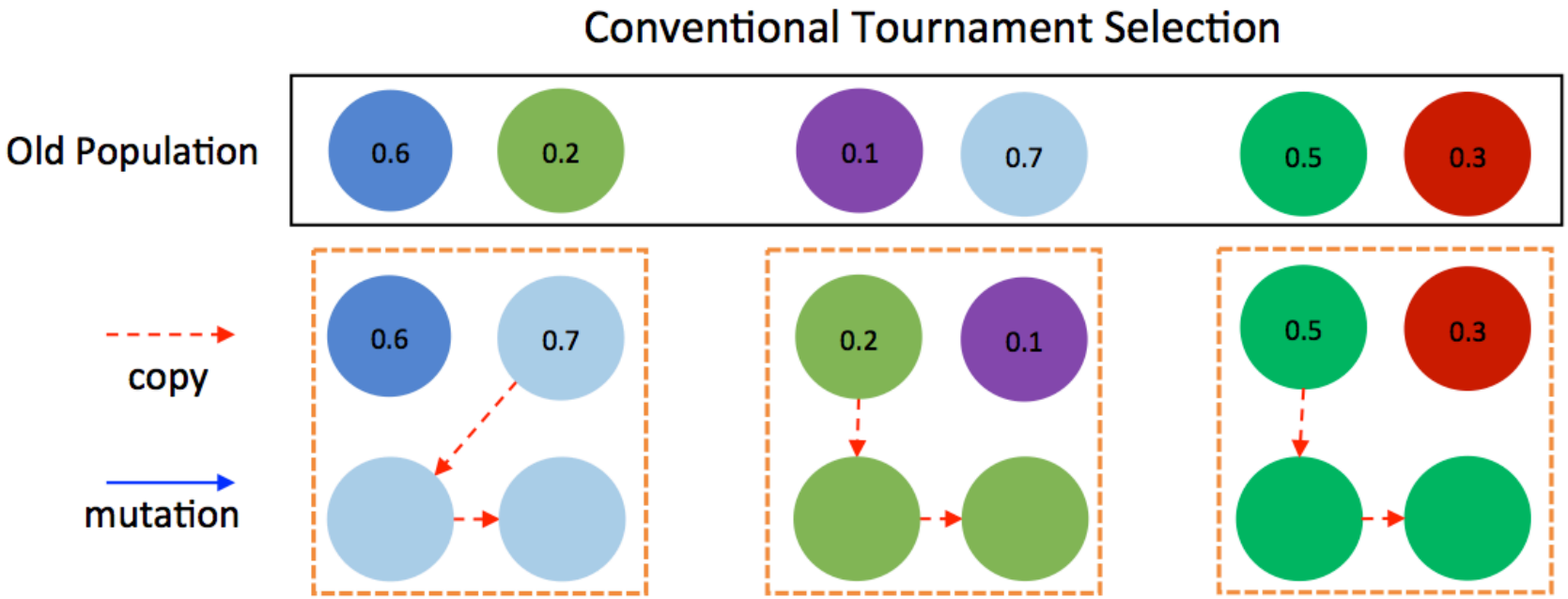
Selection Strategy - Tournament



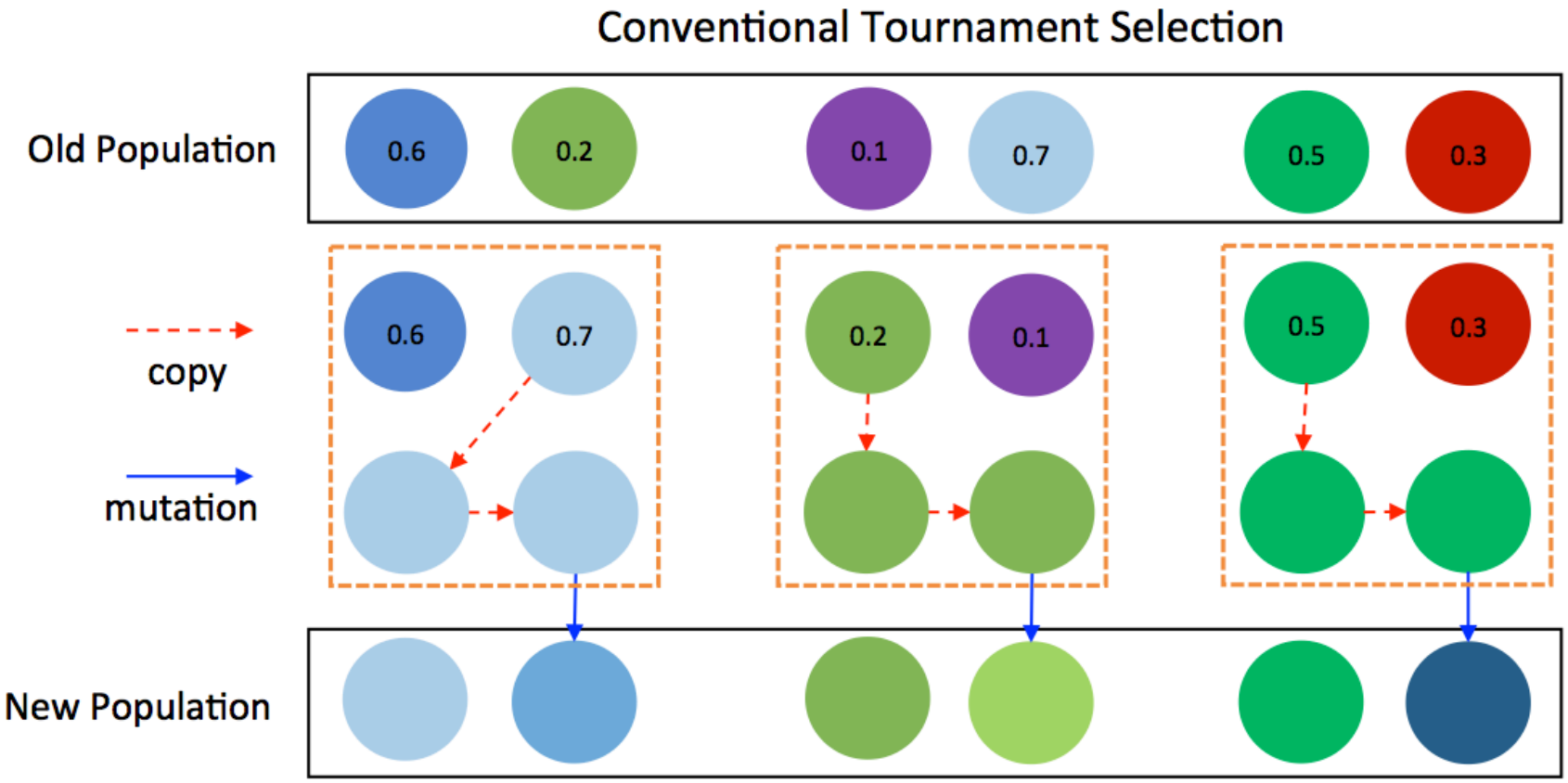
Selection Strategy - Tournament



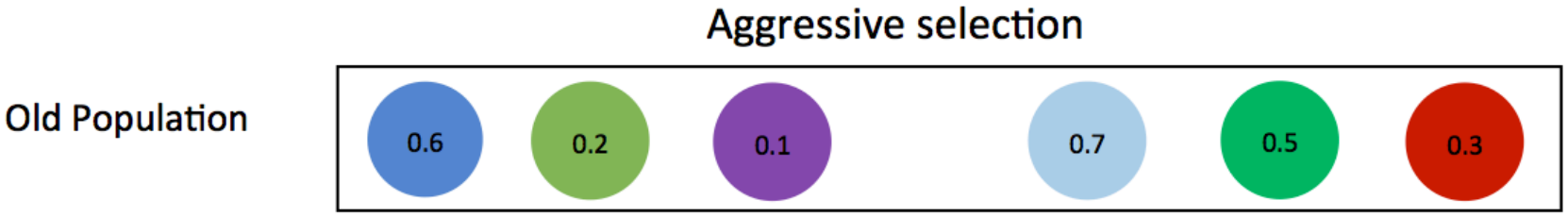
Selection Strategy - Tournament



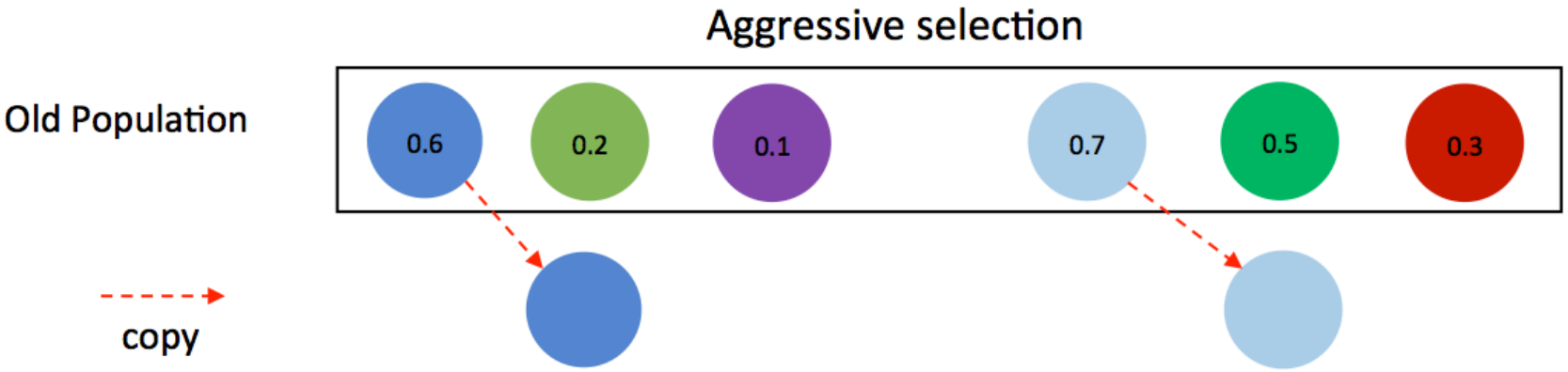
Selection Strategy - Tournament



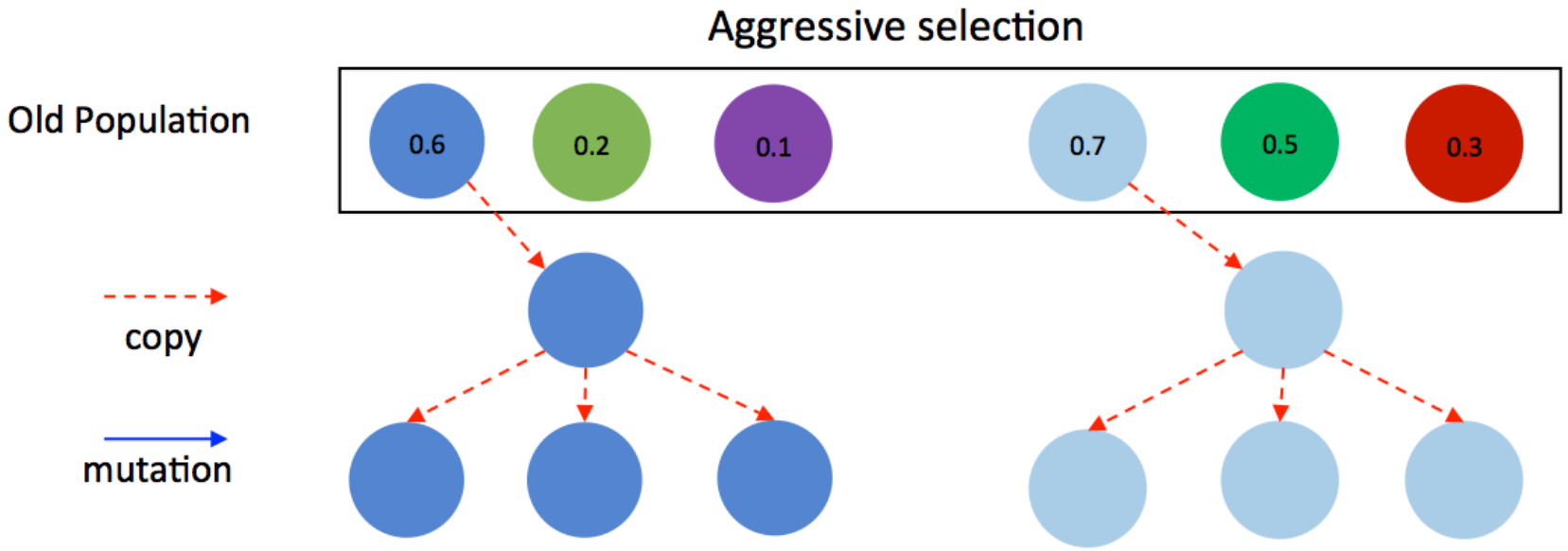
Selection Strategy – Aggressive



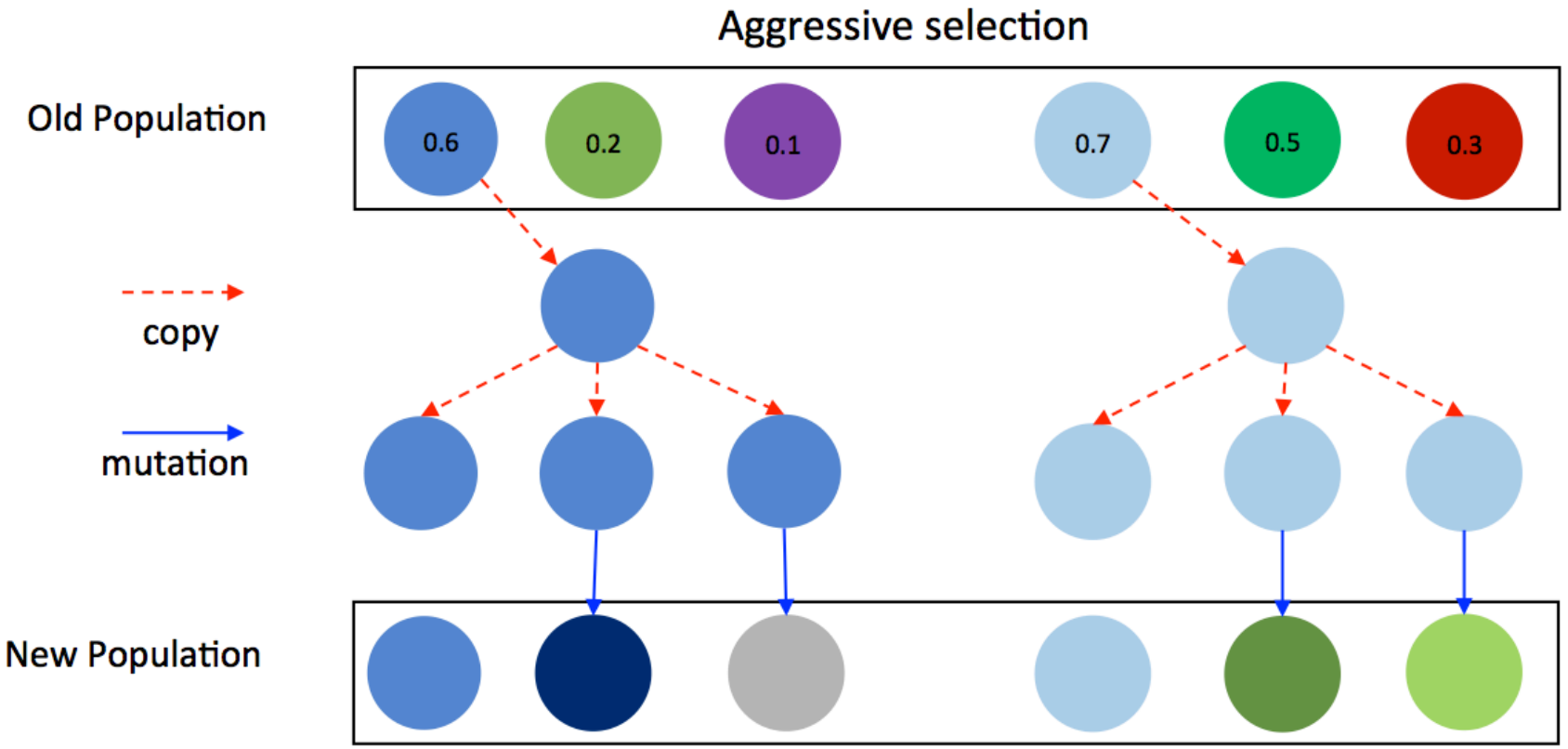
Selection Strategy – Aggressive



Selection Strategy – Aggressive



Selection Strategy – Aggressive



Selection Strategy - Aggressive

- What's the issue caused by this aggressive selection strategy compared with the three traditional strategies?
  - Decreasing the diversity of the population.

- How to increase diversity of the population?
  - More mutation operations.

More Mutation for diversity

- Mutation operations defined and implemented.

Mutations	[1]	Ours
add_convolution	✓	✓
remove_convolution	✓	✓
alter_channel_number	✓	✓
alter_filter_size	✓	✓
alter_stride	✓	✓
add_dropout	-	✓
remove_dropout	-	✓
add_pooling	-	✓
remove_pooling	-	✓
add_skip	✓	✓
remove_skip	✓	✓
add_concatenate	-	✓
remove_concatenate	-	✓
add_fully_connected	-	✓
remove_fully_connected	-	✓

The allowed mutation operations in our work and in [1]; ✓represents defined while - represents NA

[1] Large-scale Evolution of Image Classifiers

Evaluate Fitness Score

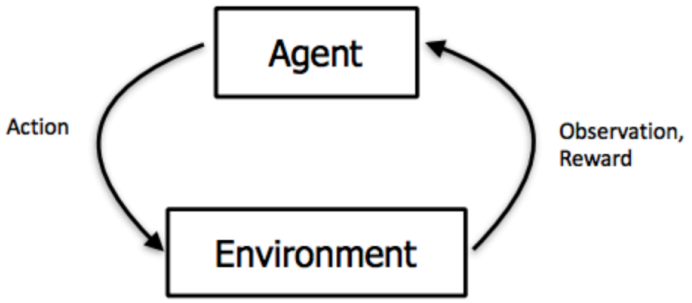
- Reducing training time
  - Early stopping – don't waste computation on the weak neural network
  - Advanced learning strategy
  - Parallel training
  - … …

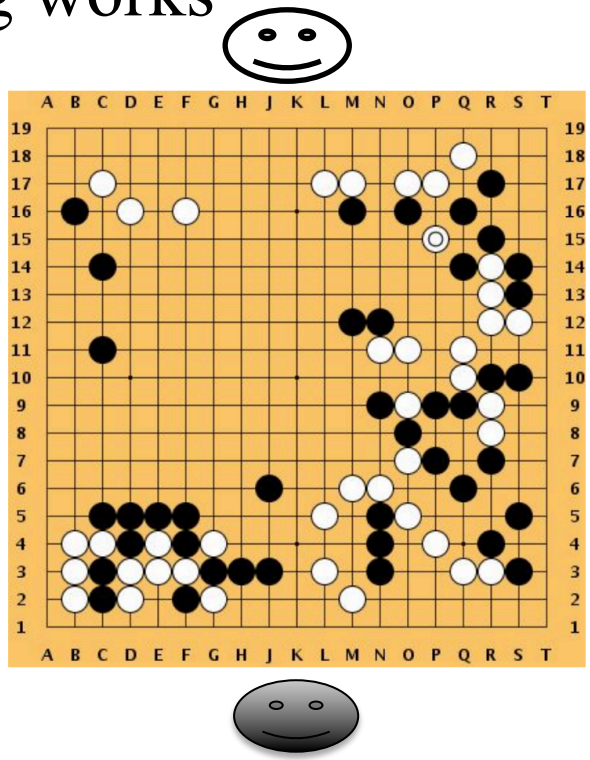
Content

- Motivation
- Genetic Approach
- **Reinforcement Learning Approach**
- Experimental Results
- Summary

Reinforcement Learning Approach

- How reinforcement learning works

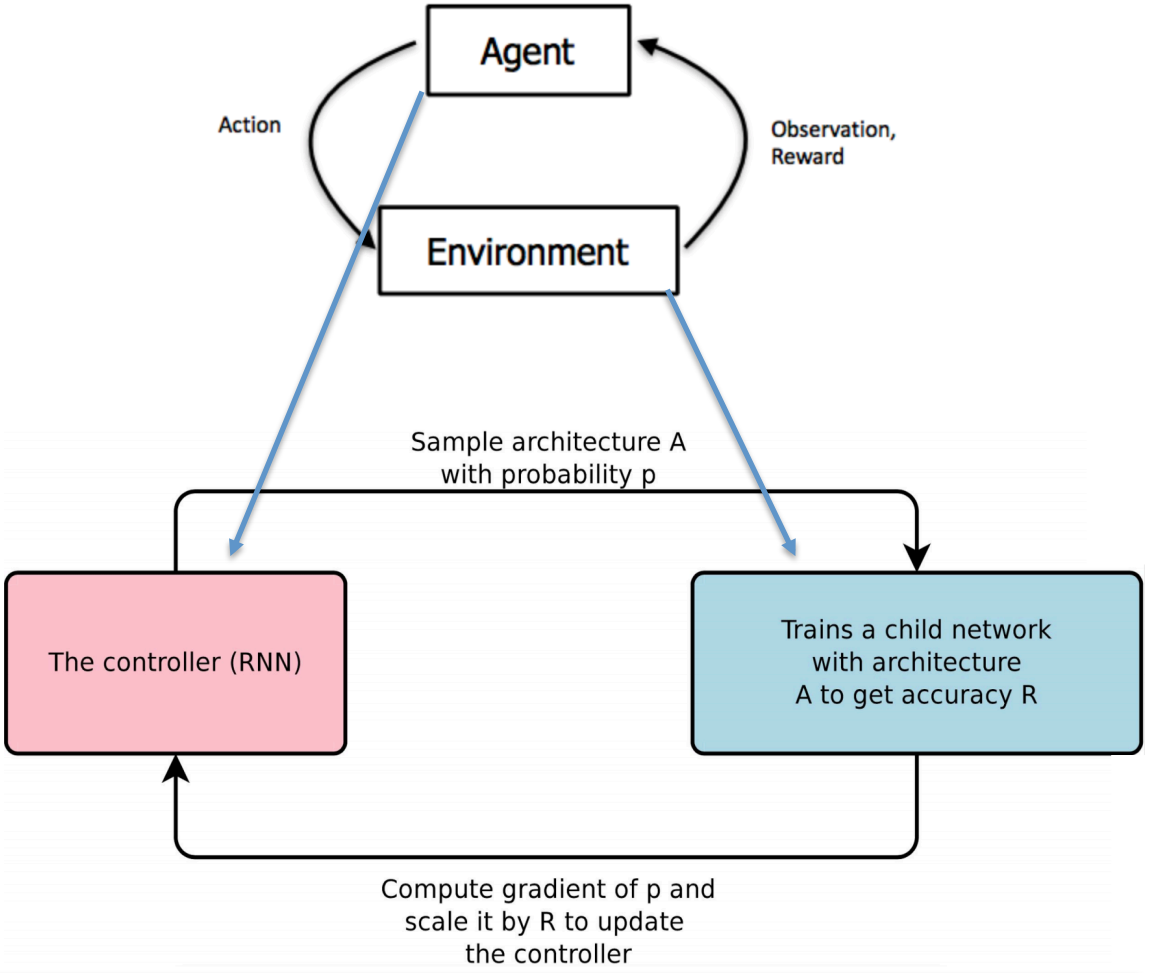




Action: where to put the next piece down.
Reward: 1 if win at the end of game, 0 otherwise.

Reinforcement Learning Approach

- **States**: Implicit states of RNN in the controller.
- **Actions**: Primitive descriptions of NN; a sequence of "actions" describes a complete NN, called "child network".
- **Rewards**: Success ratio of "child network". The controller is updated using the "rewards" by Reinforcement Learning.

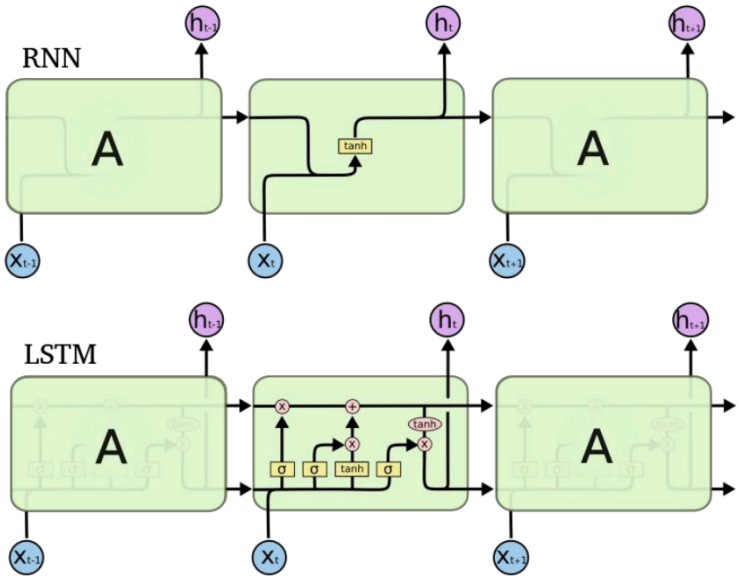


Credit: https://arxiv.org/pdf/1611.01578.pdf

Reinforcement Learning Approach

- **States**: Implicit states of RNN in the controller.
- **Actions**: Primitive descriptions of NN; a sequence of "actions" describes a complete NN, called "child network".
- **Rewards**: Success ratio of "child network". The controller is updated using the "rewards" by Reinforcement Learning.

Recurrent Architectures



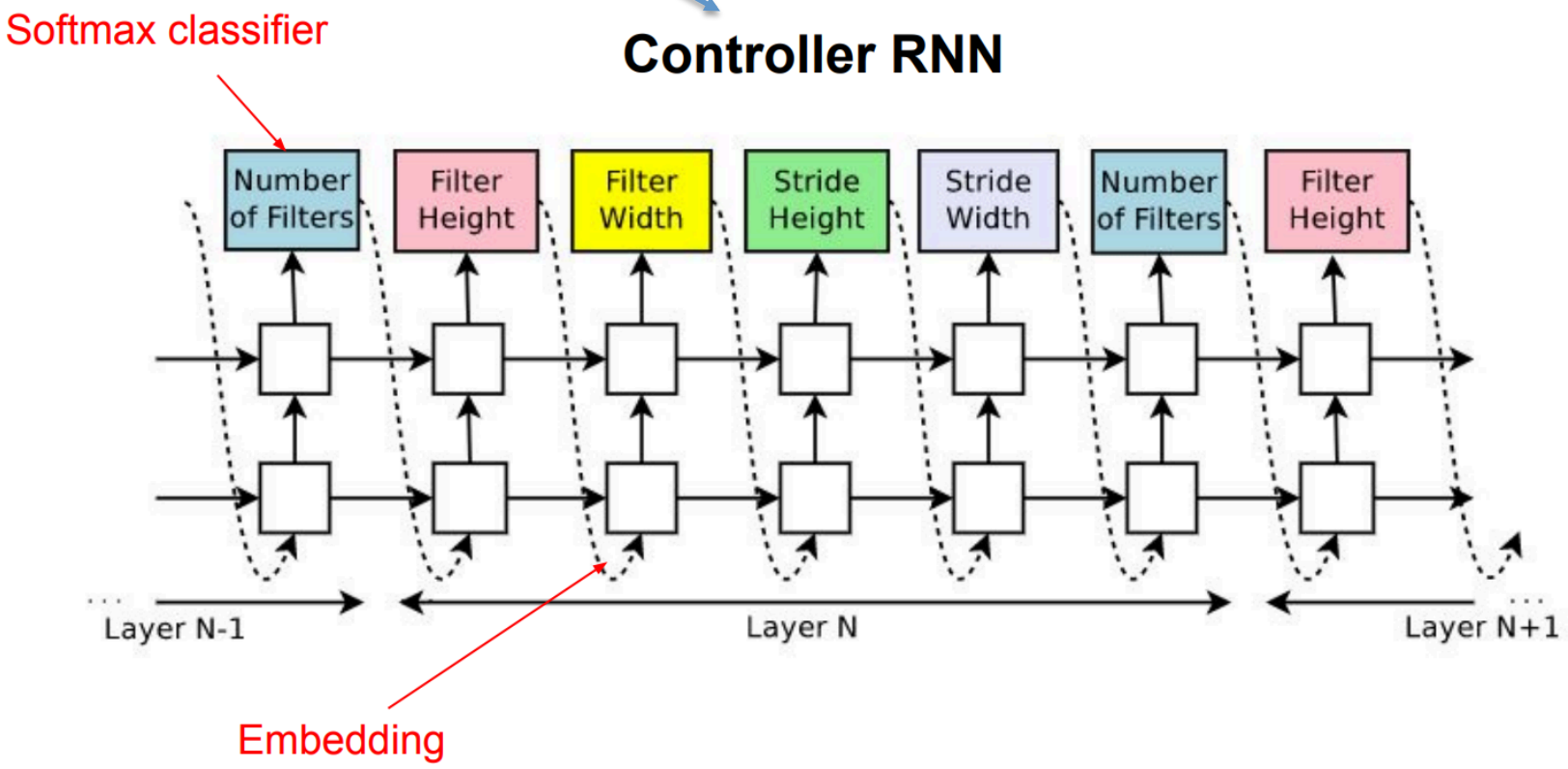
Hochreiter & Schmidhuber 1997

Credit: https://arxiv.org/pdf/1611.01578.pdf

Reinforcement Learning Approach

- **States**: Implicit states of RNN in the controller.
- **Actions**: Primitive

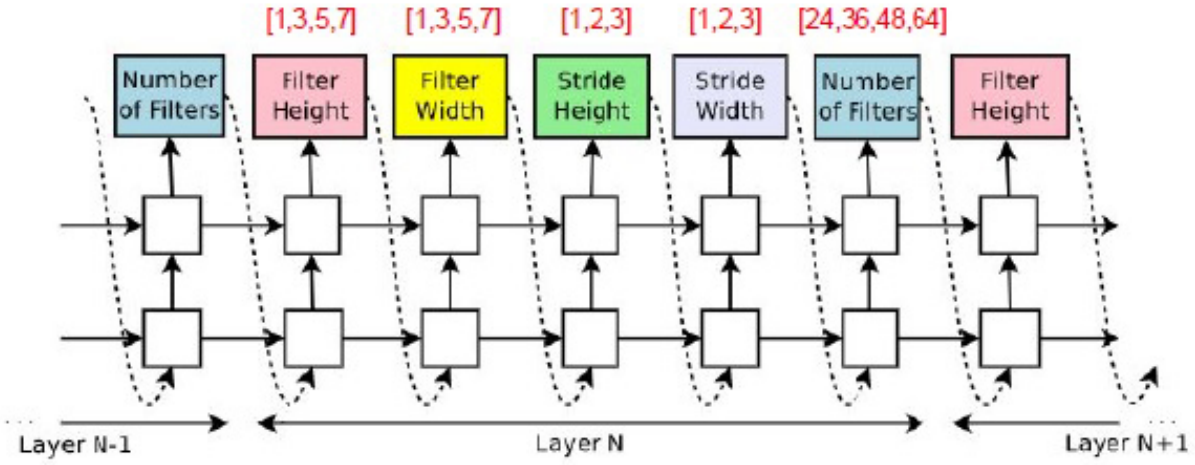
Recurrent Architectures



Reinforcement Learning Approach

- **States**: Implicit states of RNN in the controller.
- **Actions**: Primitive descriptions of NN. A sequence of "actions" describes a complete NN, called "child network".
- **Rewards**: Success ratio of "child network". The controller is updated using the "rewards" by Reinforcement Learning.

E.g., for **convolutional networks**, the primitive descriptions are "number of filters", "filter height", "filter width", "stride height", "stride width", etc. The domain of these parameters are fixed.

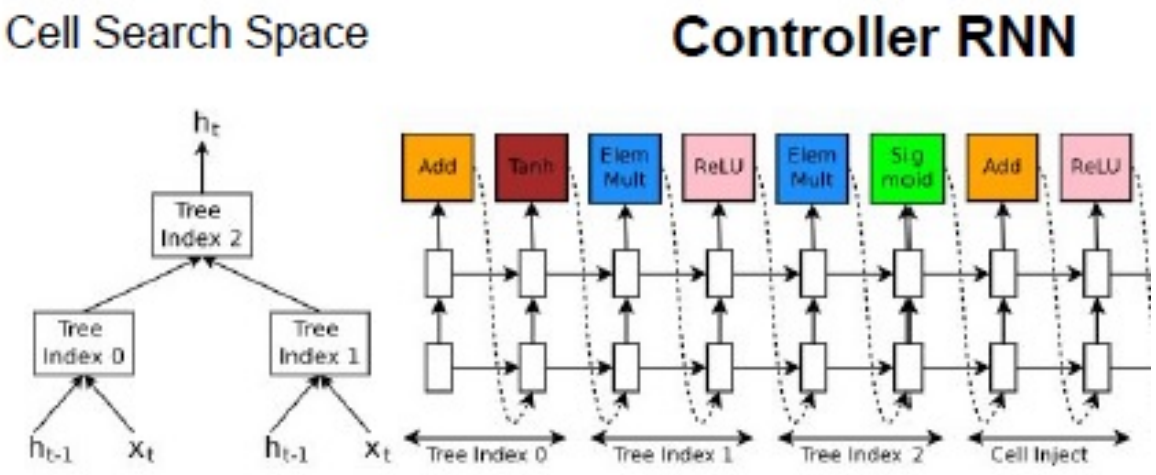


Credit: https://arxiv.org/pdf/1611.01578.pdf

Reinforcement Learning Approach

- **States**: Implicit states of RNN in the controller.
- **Actions**: Primitive descriptions of NN. A sequence of "actions" describes a complete NN, called "child network".
- **Rewards**: Success ratio of "child network". The controller is updated using the "rewards" by Reinforcement Learning.

E.g., for **the architecture of a recurrent cell**, we may fix a tree structure and define the primitive descriptions for each node in the tree as "add", "Elem mult", "Tanh", "Sigmoid", "ReLU", etc.



Credit: https://arxiv.org/pdf/1611.01578.pdf

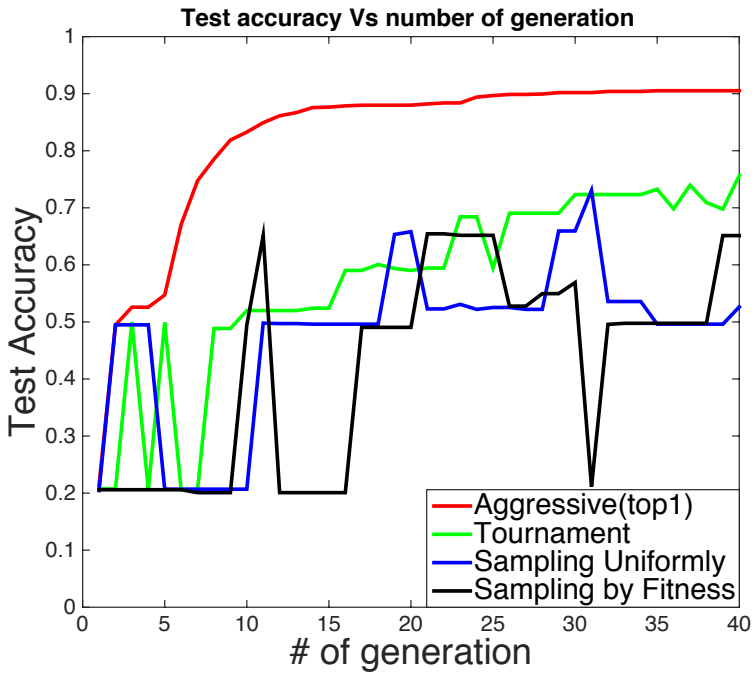
Reinforcement Learning Approach

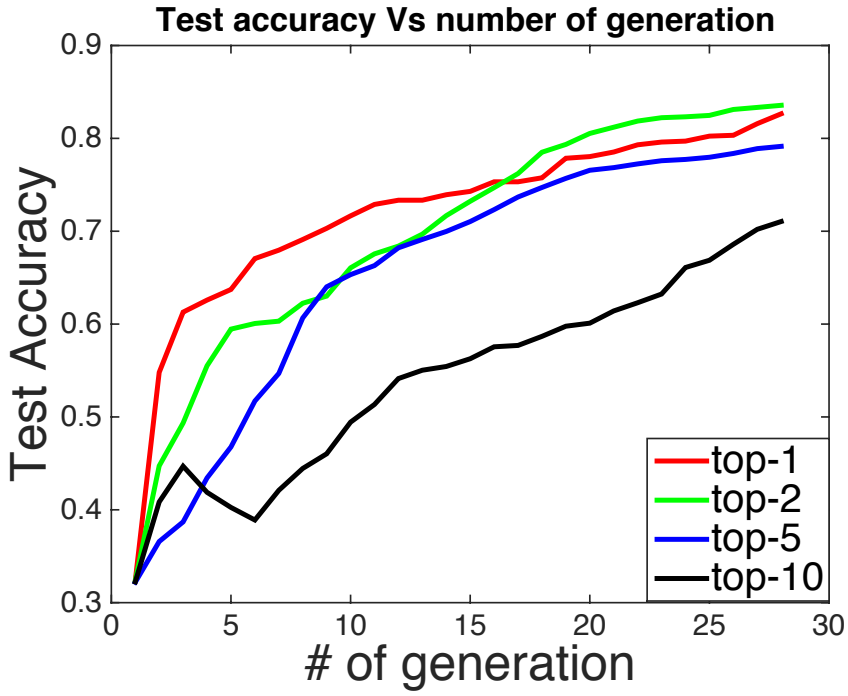
- Encode the structure and connectivity of a neural network by using a configuration string
  - ["filter width :5", "filter height:3","num filter: 24"]
- Using a RNN ("Controller") to generate this string that encodes a neural network structure
- Train this neural network structure ("child Network") to see how well it performs
- Use reinforcement learning to update the parameters of the Controller model based on the accuracy of the child model

Credit: http://rll.berkeley.edu/deeprlcoursesp17/docs/quoc_barret.pdf

Experimental Results

- Justify that the aggressive selection strategy.



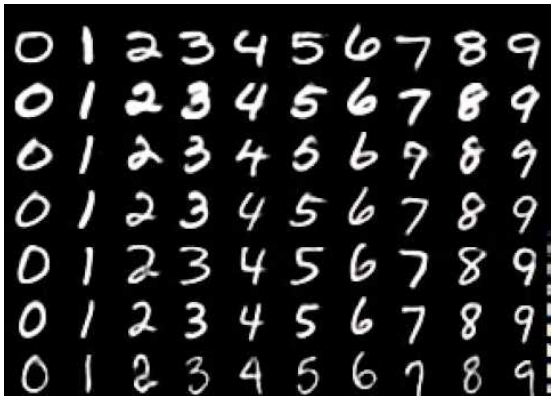


Content

- Motivation
- Genetic Approach
- Reinforcement Learning Approach
- **Experimental Results**
- Summary

Experimental Results

- Handwritten digits recognition

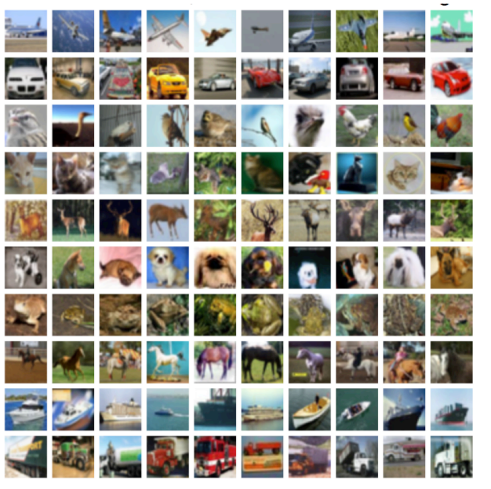


Approach	**Test Acc**	**Comp Cost**
SOTA[4]	0.9979	–
Genetic-CNN[2]	0.9966	48 GPUH
EDEN[3]	0.9840	–
Ours	0.9969	35 GPUH

Comparison of test accuracy and computational cost on MNIST dataset.

Experimental Results

- 10 Classes Image Classification

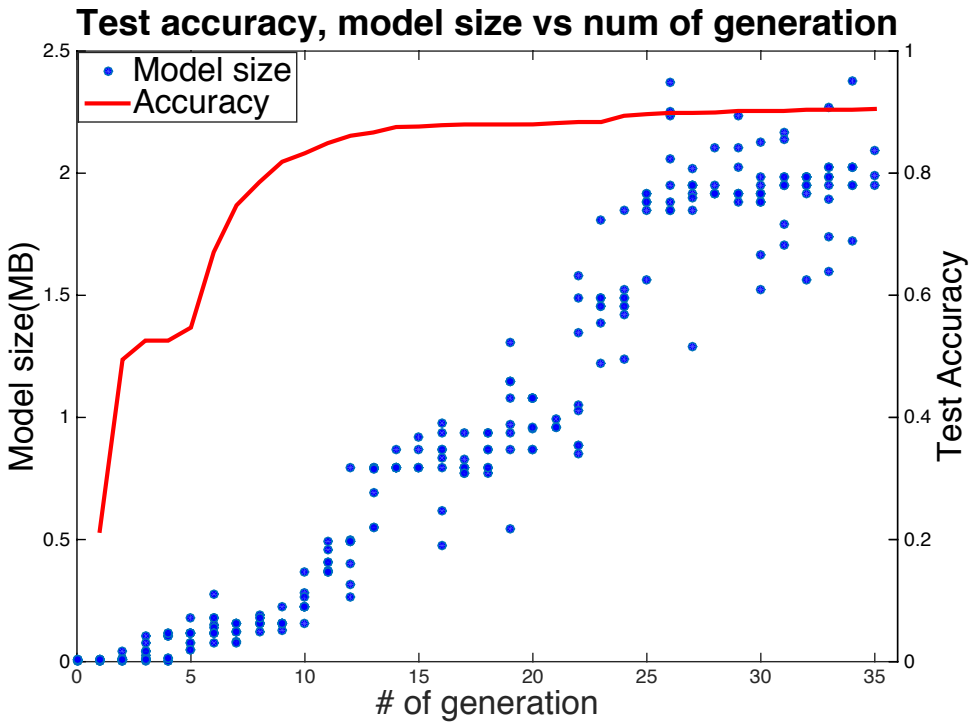


Approach	**Test Acc**	**Comp Cost**
SOTA[5]	0.9654	-
LS-Evolution[1]	0.9460	65,536 GPUH
Genetic-CNN[2]	0.7706	408 GPUH
EDEN[3]	0.7450	–
Ours	0.9267	72 GPUH

Comparison of test accuracy and computational cost on CIFAR-10

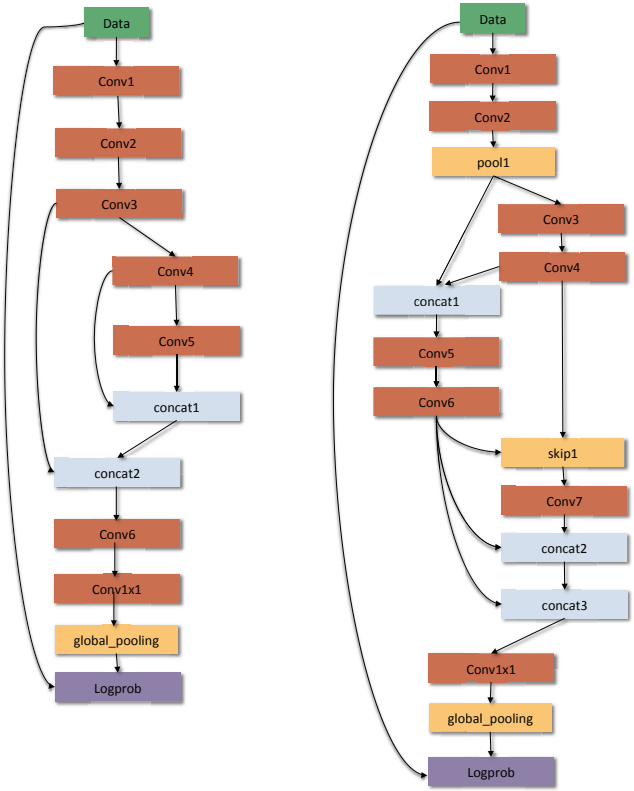
Experimental Results

- Show how model size and test accuracy of neural network change along evolution.



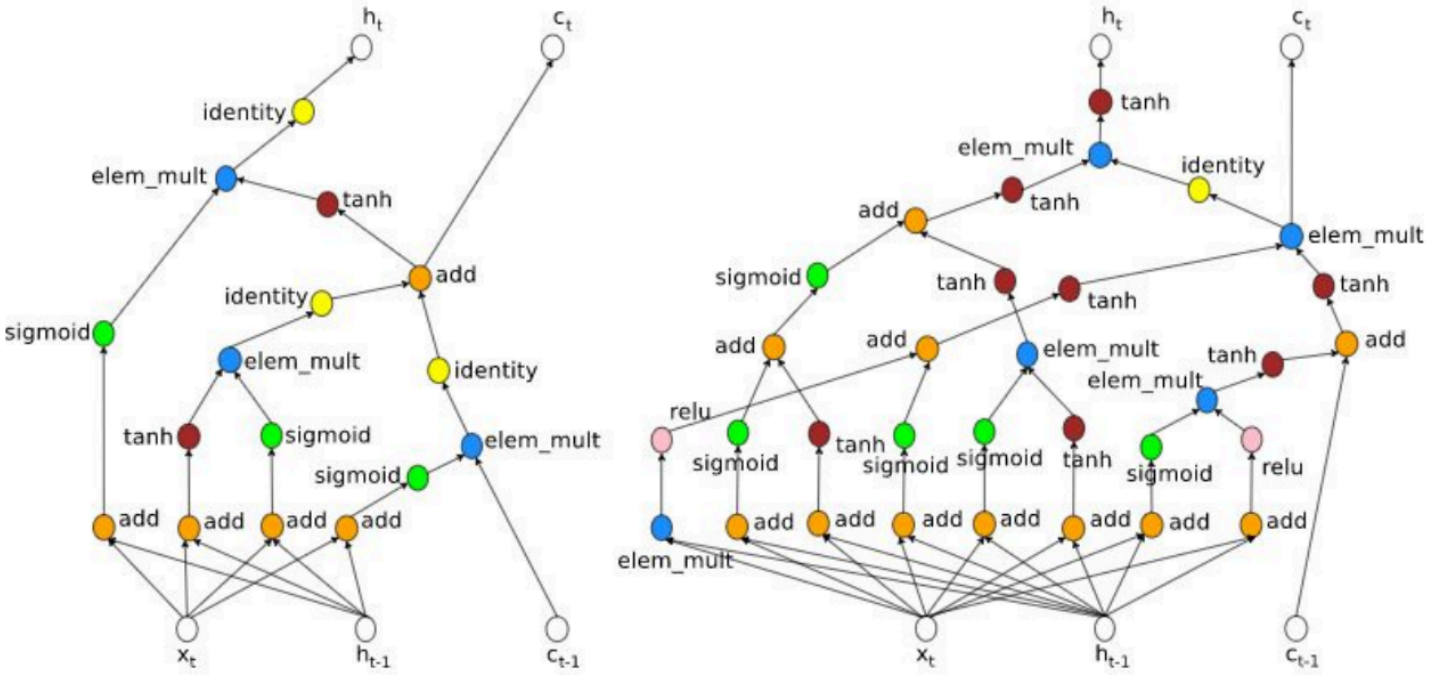
Experimental Results

- Show the finally learned neural network structures by our genetic approach.



Experimental Results

- Learned architecture of a recurrent unit by the reinforcement learning approach.



LSTM Cell

Neural Architecture Search (NAS) Cell

Content

- Motivation
- Genetic Approach
- Reinforcement Learning Approach
- Experimental Results
- **Summary**

Summary

- Automatically searching neural network structures is hot area.
- Explored genetic approach to search neural network.
- Implemented genetic framework for this task.
- Proposed different strategies to reduce computation cost.
- Briefly discussed reinforcement learning approach to search neural network structures.
- Achieved competitive performance with significantly reduced computational cost by genetic approach.