# AutoML: Neural Architecture Search (NAS)

## Blackbox Optimization Methods

Bernd Bischl   Frank Hutter   Lars Kotthoff
Marius Lindauer   Joaquin Vanschoren

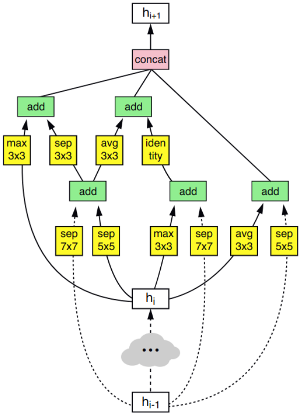
# NAS as Hyperparameter Optimization

- NAS can be formulated as a HPO problem

# NAS as Hyperparameter Optimization

- NAS can be formulated as a HPO problem

- E.g., cell search space by [Zoph et al. 2018] has 5 categorical choices per block
  - 2 categorical choices of hidden states
    - For block $N$, the domain of these categorical variables is $\{h_i, h_{i-1}, \text{output of block } 1, ..., \text{output of block } N-1\}$
  - 2 categorical variables choosing between operations
  - 1 categorical variable choosing the combination method
  - Total number of hyperparameters for the cell: 5B (with B=5 by default)

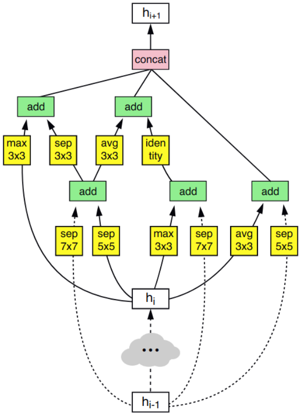


*Reduction Cell*

# NAS as Hyperparameter Optimization

- NAS can be formulated as a HPO problem

- E.g., cell search space by [Zoph et al. 2018] has 5 categorical choices per block
  - 2 categorical choices of hidden states
    - For block $N$, the domain of these categorical variables is $\{h_i, h_{i-1}, \text{output of block } 1, ..., \text{output of block } N-1\}$
  - 2 categorical variables choosing between operations
  - 1 categorical variable choosing the combination method
  - Total number of hyperparameters for the cell: 5B (with B=5 by default)




Reduction Cell

- In general: one may require conditional hyperparameters
  - E.g., chain-structured search space
    - Top-level hyperparameter: number of layers L
    - Hyperparameters of layer k conditional on L $\geq$ k

# Early Work on Neuroevolution (already since the 1990s)

[Kitano. 1990; Angeline et al. 1994; Stanley and Miikkulainen. 2002; Bayer et al. 2009; Floreano et al. 2008]

- Evolves architectures & often also their weights

# Early Work on Neuroevolution (already since the 1990s)

[Kitano. 1990; Angeline et al. 1994; Stanley and Miikkulainen. 2002; Bayer et al. 2009; Floreano et al. 2008]

- Evolves architectures & often also their weights
- Typical approach:
  - Initialize a population of $N$ random architectures
  - Sample $N$ individuals from that population (with replacement) according to their fitness
  - Apply mutations to those $N$ individuals to produce the next generation's population
  - Optionally: elitism to keep best individuals in the population

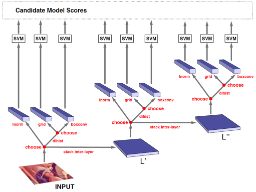
# Early Work on Neuroevolution (already since the 1990s)

[Kitano. 1990; Angeline et al. 1994; Stanley and Miikkulainen. 2002; Bayer et al. 2009; Floreano et al. 2008]

- Evolves architectures & often also their weights
- Typical approach:
  - Initialize a population of $N$ random architectures
  - Sample $N$ individuals from that population (with replacement) according to their fitness
  - Apply mutations to those $N$ individuals to produce the next generation's population
  - Optionally: elitism to keep best individuals in the population
- Mutations include adding, changing or removing a layer

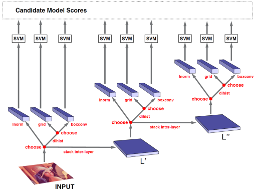
# Early Work on Bayesian Optimization (since 2013)

- With TPE [Bergstra et al. 2011]:
  - Joint optimization of a vision architecture with 238 hyperparameters [Bergstra et al. 2013]
  - State-of-the-art performance on 3 disparate problems:
    - Face matching, face identification, and object recognition



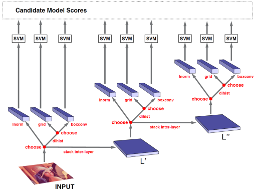
# Early Work on Bayesian Optimization (since 2013)

- With TPE [Bergstra et al. 2011]:
  - Joint optimization of a vision architecture with 238 hyperparameters [Bergstra et al. 2013]
  - State-of-the-art performance on 3 disparate problems:
    - Face matching, face identification, and object recognition
- With SMAC [Hutter et al. 2011]:
  - New state-of-the-art performance on CIFAR-10 w/o data augmentation [Domhan et al. 2015]
  - Joint architecture and hyperparameter search, yielding Auto-Net [Mendoza et al. 2016]

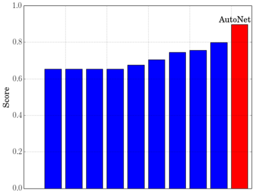


# Early Work on Bayesian Optimization (since 2013)

- With TPE [Bergstra et al. 2011]:
  - Joint optimization of a vision architecture with 238 hyperparameters [Bergstra et al. 2013]
  - State-of-the-art performance on 3 disparate problems:
    - Face matching, face identification, and object recognition

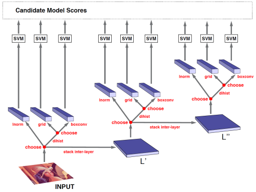


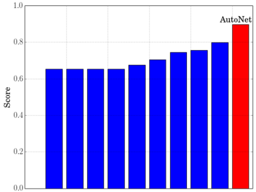

- With SMAC [Hutter et al. 2011]:
  - New state-of-the-art performance on CIFAR-10 w/o data augmentation [Domhan et al. 2015]
  - Joint architecture and hyperparameter search, yielding Auto-Net [Mendoza et al. 2016]
  - In 2015, Auto-Net already had several successes in ML competitions
    - E.g., human action recognition:
      54491 data points, 5000 features, 18 classes
    - First automated deep learning (Auto-DL) method to win a machine learning competition dataset against human experts



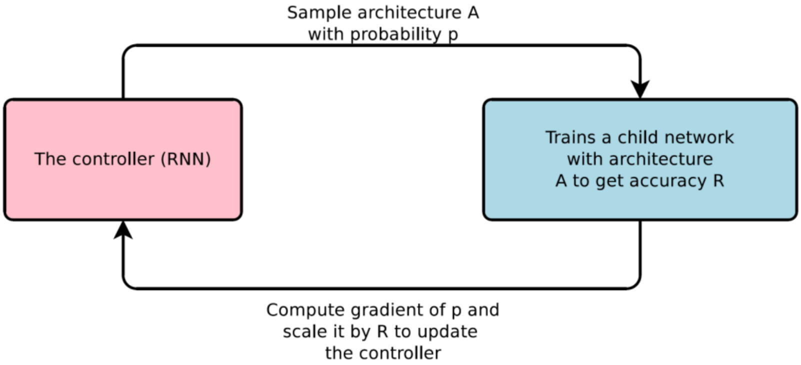
# Early Work on Bayesian Optimization (since 2013)

- With TPE [Bergstra et al. 2011]:
  - Joint optimization of a vision architecture with 238 hyperparameters [Bergstra et al. 2013]
  - State-of-the-art performance on 3 disparate problems:
    - Face matching, face identification, and object recognition
- With SMAC [Hutter et al. 2011]:
  - New state-of-the-art performance on CIFAR-10 w/o data augmentation [Domhan et al. 2015]
  - Joint architecture and hyperparameter search, yielding Auto-Net [Mendoza et al. 2016]
  - In 2015, Auto-Net already had several successes in ML competitions
    - E.g., human action recognition:
      54491 data points, 5000 features, 18 classes
    - First automated deep learning (Auto-DL) method to win a machine learning competition dataset against human experts
- With Gaussian processes:
  - Arc kernel [Swersky et al. 2013]



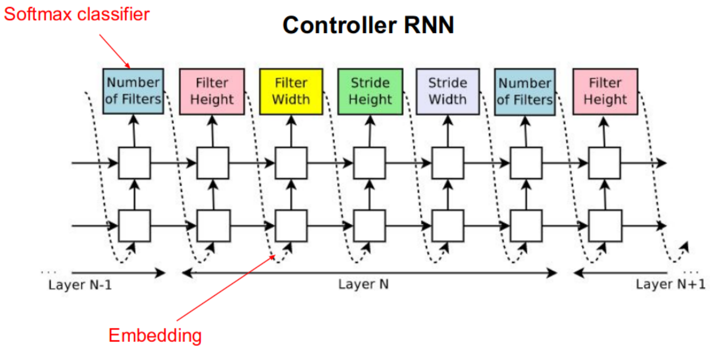



# Reinforcement Learning [Zoph and Le. 2016]



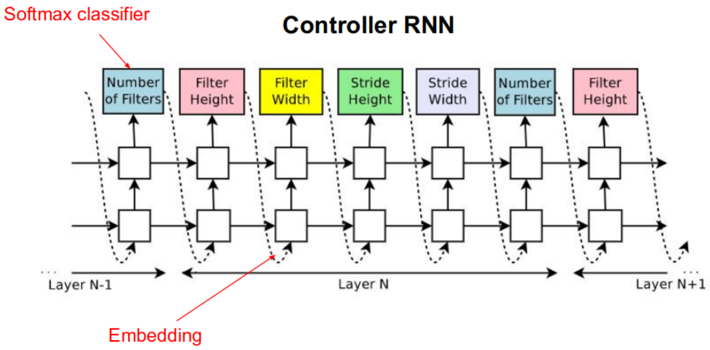

- Use RNN ("Controller") to generate a NN architecture piece-by-piece
- Train this NN ("Child Network") and evaluate it on a validation set
- Use Reinforcement Learning (RL) to update the parameters of the Controller RNN to optimize the performance of the child models

# Learning CNNs with RL [Zoph and Le. 2016]



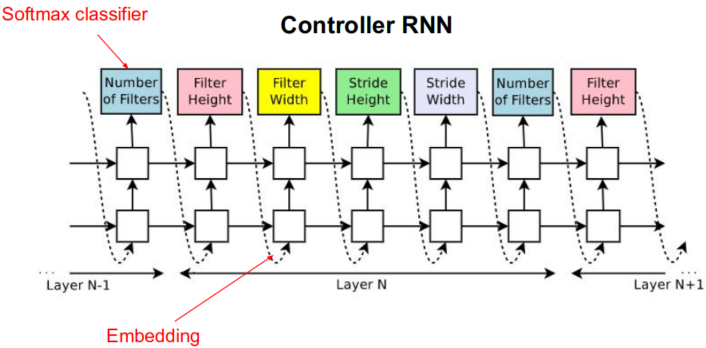

- For a fixed number of layers, select:
  - Filter width/height, stride width/height, number of filters

# Learning CNNs with RL [Zoph and Le. 2016]




- For a fixed number of layers, select:
  - Filter width/height, stride width/height, number of filters
- Large computational demands (800 GPUs for 2 weeks, 12.800 architectures evaluated)

# Learning CNNs with RL [Zoph and Le. 2016]

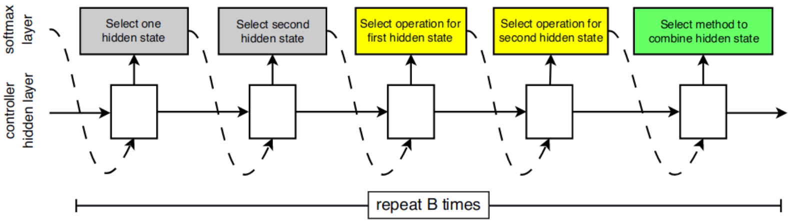


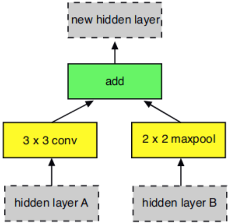

- For a fixed number of layers, select:
  - Filter width/height, stride width/height, number of filters
- Large computational demands (800 GPUs for 2 weeks, 12.800 architectures evaluated)
- State-of-the-art results for CIFAR-10 & Penn Treebank architecture

  $\rightarrow$ Brought NAS into the limelight

# Learning CNN cells with RL [Zoph et al. 2018]

- 2 types of cells: normal and reduction cells
- For each type of cell: $B$ blocks, each with 5 choices
  - Choose two previous feature maps (from this cell)
  - For each of these, choose an operation (3$\times$3 conv, max-pool, etc.)
  - Choose a merge operation to combine the two results (concat or add)

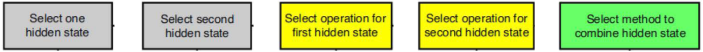


# Learning CNN cells with evolution [Real et al. 2018]

- 2 types of cells: normal and reduction cells
- For each type of cell: $B$ blocks, each with 5 choices
  - Choose two previous feature maps (from this cell)
  - For each of these, choose an operation ($3\times3$ conv, max-pool, etc.)
  - Choose a merge operation to combine the two results (concat or add)

- Evolution simply tackles this as a HPO problem with $2\times5\times B$ variables:

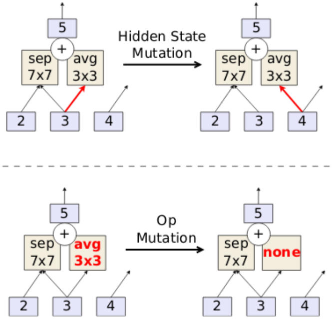


# Regularized/Aging Evolution [Real et al. 2018]

- Quite standard evolutionary algorithm
  - But oldest solutions are dropped from population, instead of the worst

# Regularized/Aging Evolution [Real et al. 2018]

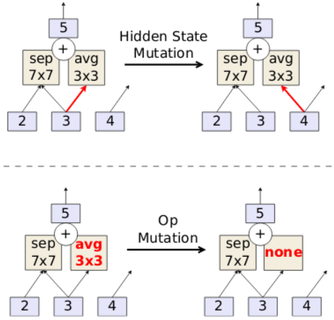
- Quite standard evolutionary algorithm
  - But oldest solutions are dropped from population, instead of the worst
- Standard SGD for training weights (optimizing the same blackbox as RL)
- Same fixed-length (HPO) search space as used for RL [Zoph et al. 2018]



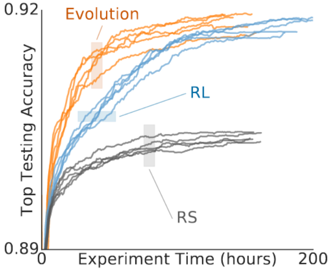

Different types of mutations in cell search space

# Regularized/Aging Evolution [Real et al. 2018]

- Quite standard evolutionary algorithm
  - But oldest solutions are dropped from population, instead of the worst
- Standard SGD for training weights (optimizing the same blackbox as RL)
- Same fixed-length (HPO) search space as used for RL [Zoph et al. 2018]




Different types of mutations in cell search space




State-of-the-art performance in apples-to-apples comparison
→ AmoebaNet

# Bayesian Optimization (BO)

- Encode the architecture space by categorical hyperparameters (like regularized evolution)
- Strong performance with tree-based models
  - TPE [Bergstra et al. 2013]
  - SMAC [Domhan et al. 2015; Mendoza et al. 2016; Zela et al. 2018]

# Bayesian Optimization (BO)

- Encode the architecture space by categorical hyperparameters (like regularized evolution)
- Strong performance with tree-based models
  - TPE [Bergstra et al. 2013]
  - SMAC [Domhan et al. 2015; Mendoza et al. 2016; Zela et al. 2018]
- Kernels for GP-based NAS
  - Arc kernel [Swersky et al. 2013]
  - NASBOT [Kandasamy et al. 2018]

# Bayesian Optimization (BO)

- Encode the architecture space by categorical hyperparameters (like regularized evolution)
- Strong performance with tree-based models
  - TPE [Bergstra et al. 2013]
  - SMAC [Domhan et al. 2015; Mendoza et al. 2016; Zela et al. 2018]
- Kernels for GP-based NAS
  - Arc kernel [Swersky et al. 2013]
  - NASBOT [Kandasamy et al. 2018]
- There are also several recent promising BO approaches based on neural networks
  - BANANAS [White et al. 2019]

# Bayesian Optimization (BO)

- Encode the architecture space by categorical hyperparameters (like regularized evolution)
- Strong performance with tree-based models
  - TPE [Bergstra et al. 2013]
  - SMAC [Domhan et al. 2015; Mendoza et al. 2016; Zela et al. 2018]
- Kernels for GP-based NAS
  - Arc kernel [Swersky et al. 2013]
  - NASBOT [Kandasamy et al. 2018]
- There are also several recent promising BO approaches based on neural networks
  - BANANAS [White et al. 2019]
- BO is very competitive, has been shown to outperform RL [Ying et al. 2019]

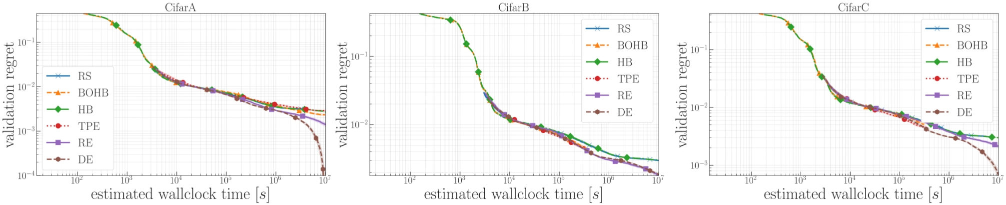
# Current State of the Art: Differential Evolution

- Comprehensive experiments on a wide range of 12 different NAS benchmarks
  [Awad et al. 2020]

# Current State of the Art: Differential Evolution

- Comprehensive experiments on a wide range of 12 different NAS benchmarks [Awad et al. 2020]

- Results:
  - Regularized evolution is very robust, typically amongst best of the methods discussed so far
  - Evolution variant of differential evolution is yet better; most efficient and robust method



# Questions to Answer for Yourself / Discuss with Friends

- Repetition:
  What are some pros and cons of using black-box optimizers for NAS?
- Repetition:
  How can NAS be modelled as a HPO problem?
- Discussion:
  Given enough resources, will blackbox NAS approaches always improve performance?
- Discussion:
  Why does discarding the oldest individual (rather than the worst) help in regularized/aging evolution?
- Transfer:
  How would you write NAS with the hierarchical search space as a HPO problem?