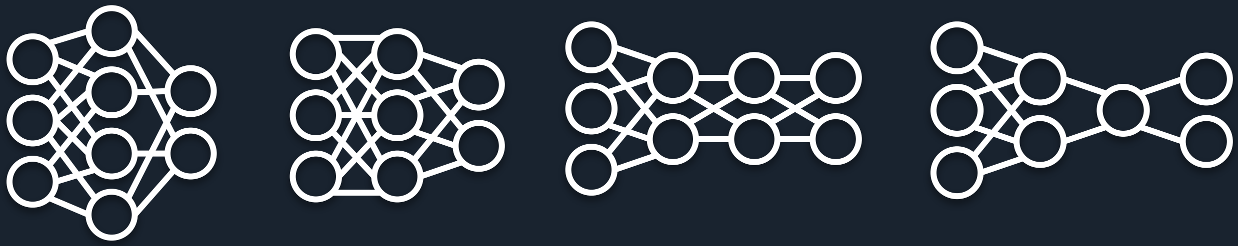
An Introduction to Neural Architecture Search

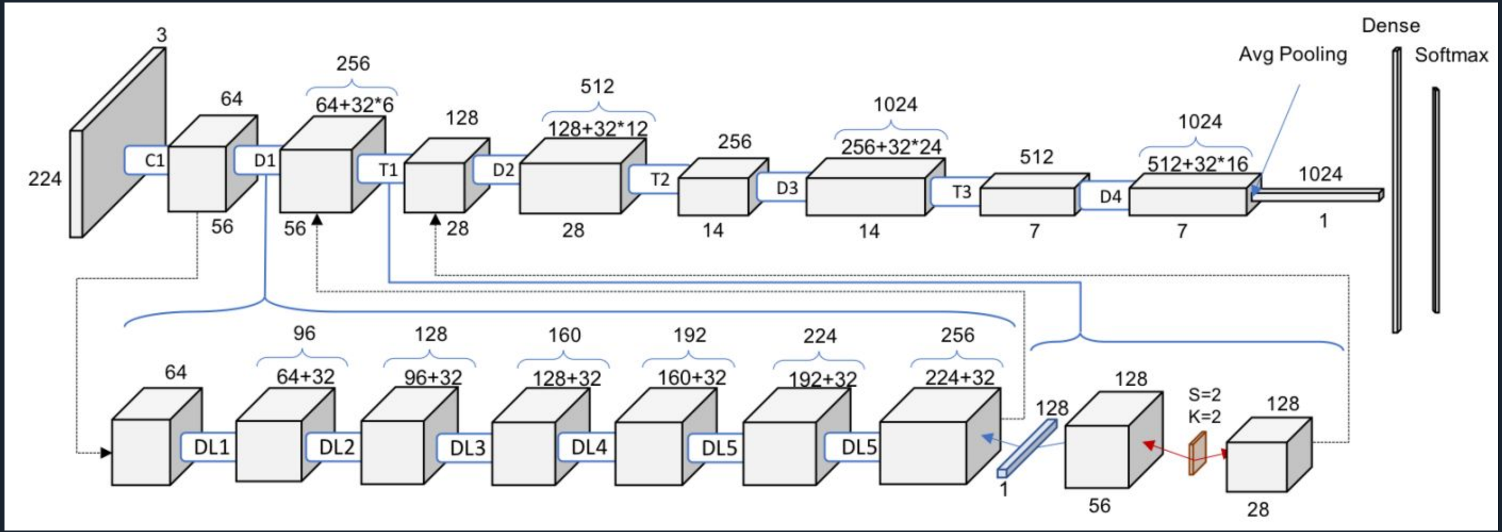
Colin White, RealityEngines.AI

Deep learning

- Explosion of interest since 2012
- Very powerful machine learning technique
- Huge variety of neural networks for different tasks
- Key ingredients took years to develop
- Algorithms are getting increasingly more specialized and complicated



Deep learning

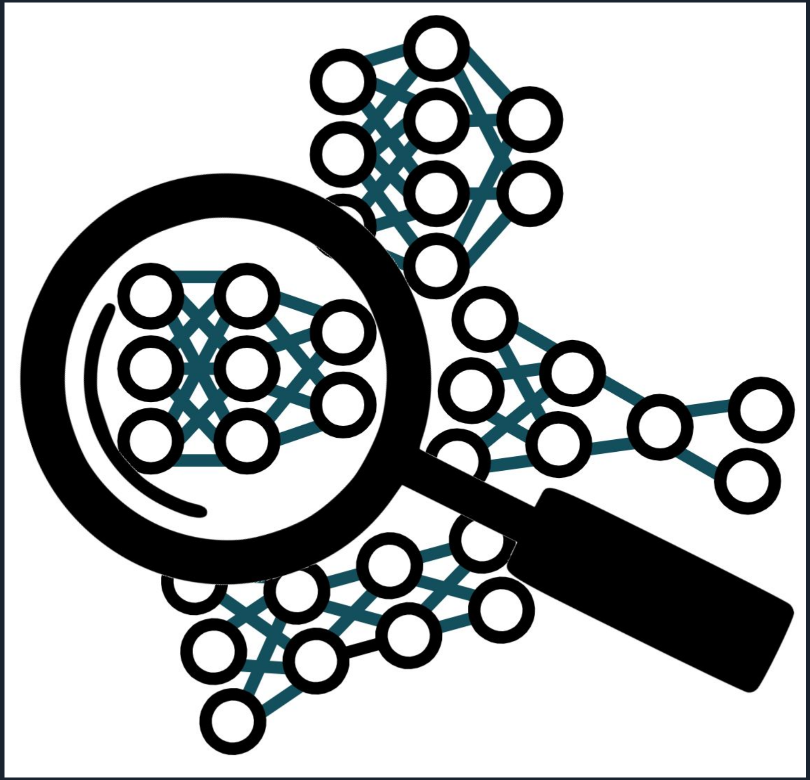


Algorithms are getting increasingly more specialized and complicated

E.g. accuracy on ImageNet has steadily improved over 10 years

Neural architecture search

- What if an algorithm could do this for us?
- Neural architecture search (NAS) is a hot area of research
- Given a dataset, define a search space of architectures, then use a search strategy to find the best architecture for your dataset

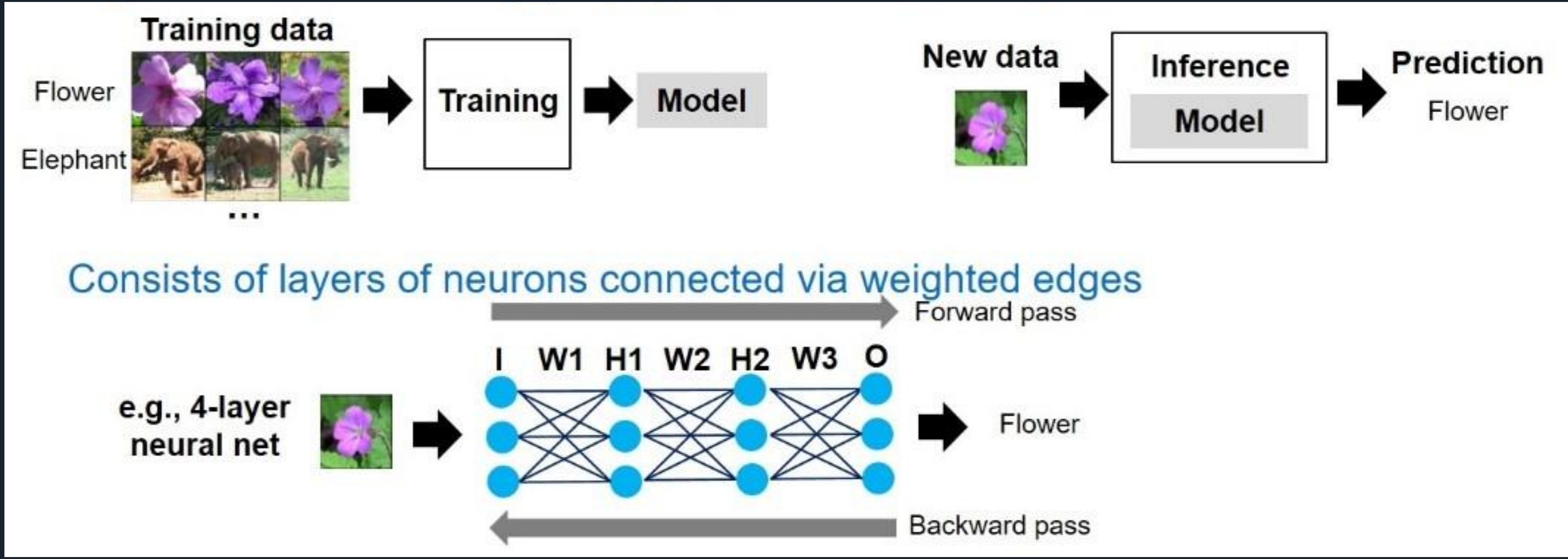


Outline

- Introduction to NAS
- Background on deep learning
- Automated machine learning
- Optimization techniques
- NAS Framework
  - Search space
  - Search strategy
  - Evaluation method
- Conclusions

Background - deep learning

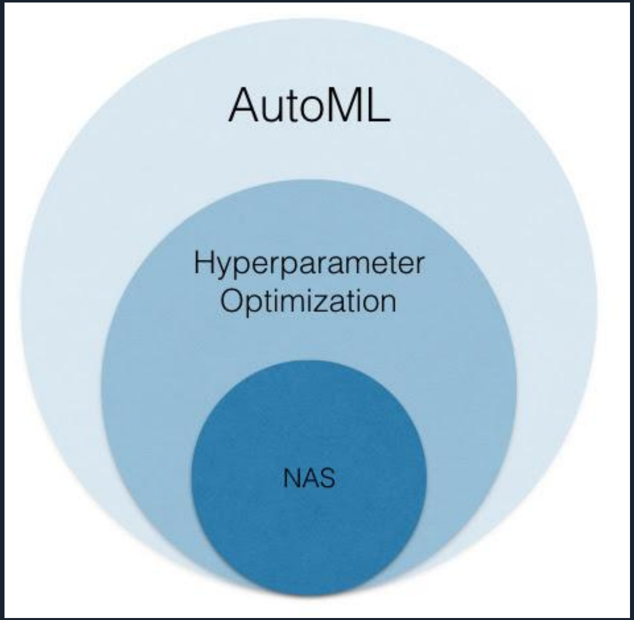
- Studied since the 1940s - simulate the human brain
- “Neural networks are the second-best way to do almost anything” - JS Denker, 2000s
- Breakthrough: 2012 ImageNet competition [Krizhevsky, Sutskever, and Hinton]



Source:https://www.nextplatform.com/2017/03/21/can-fpgas-beat-gpus-accelerating-next-generation-deep-learning/

Automated Machine Learning

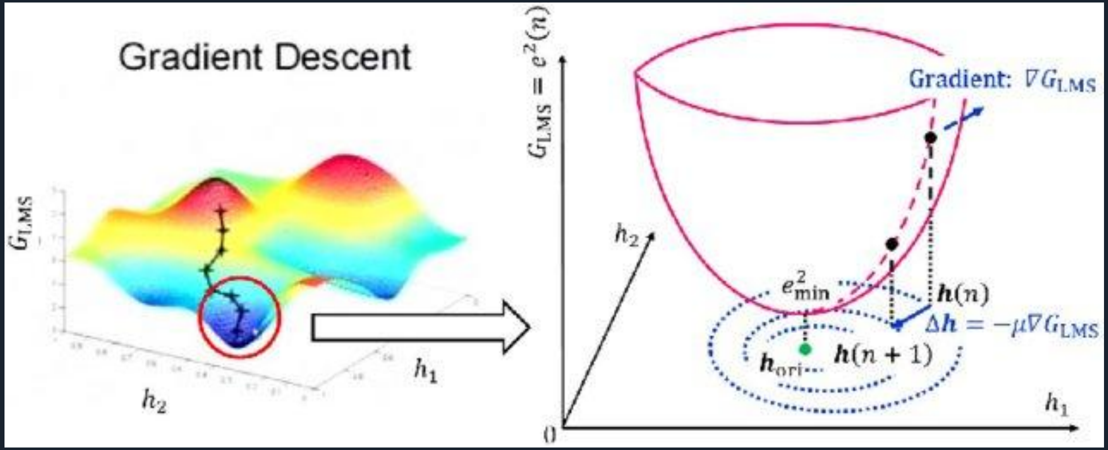
- Automated machine learning
  - Data cleaning, model selection, HPO, NAS, ...
- Hyperparameter optimization (HPO)
  - Learning rate, dropout rate, batch size, ...
- Neural architecture search
  - Finding the best neural architecture



Source:https://determined.ai/blog/neural-architecture-search/

Optimization

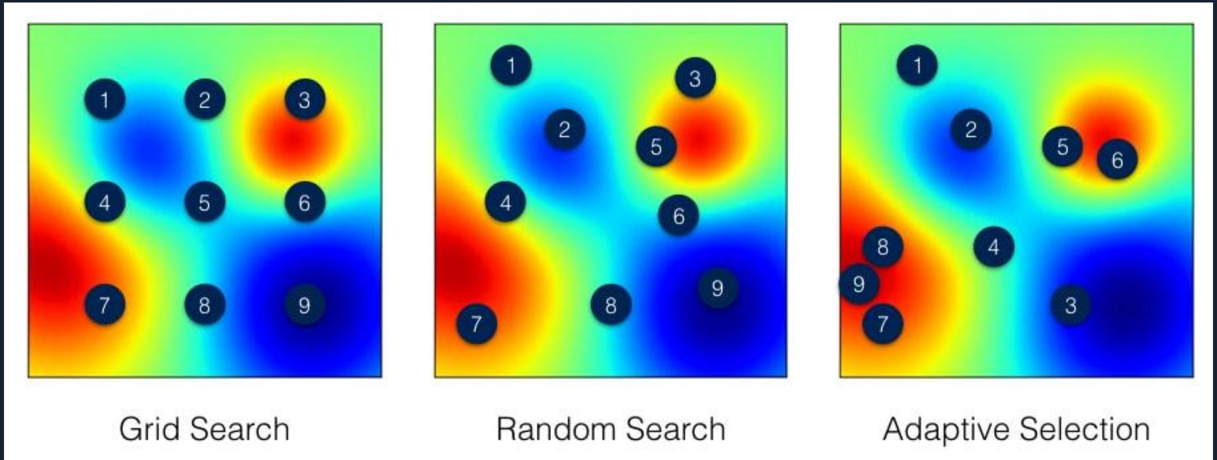
- Zero'th order optimization (used for HPO, NAS)
    - Bayesian Optimization
- First order optimization (used for Neural nets)
    - Gradient descent / stochastic gradient descent
- Second order optimization
    - Newton's method



Source: https://ieeexplore.ieee.org/document/8422997

Zero'th order optimization

- Grid search
- Random search
- Bayesian Optimization
  - Use the results of the previous guesses to make the next guess



Source: https://blog.ml.cmu.edu/2018/12/12/massively-parallel-hyperparameter-optimization/

Outline

- Introduction to NAS
- Background on deep learning
- Automated machine learning
- Optimization techniques
- NAS Framework
  - Search space
  - Search strategy
  - Evaluation method
- Conclusions

Neural Architecture Search

Search space

- Cell-based search space
- Macro vs micro search

Search strategy

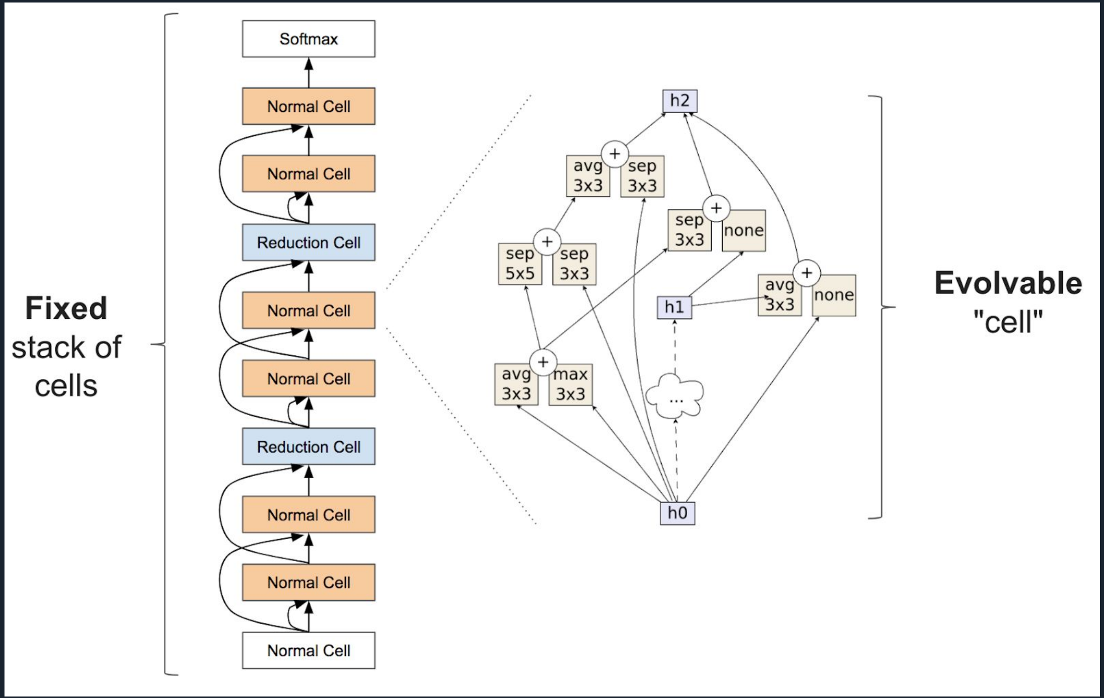
- Reinforcement learning
- Continuous methods
- Bayesian optimization
- Evolutionary algorithm

Evaluation method

- Full training
- Partial training
- Training with shared weights

Search space

- Macro vs micro search
- Progressive search
- Cell-based search space



[Zoph, Le ‘16]

Bayesian optimization

- Popular method [Golovin et al. ‘17], [Jin et al. ‘18], [Kandasamy et al. ‘18]
- Great method to optimize an expensive function

- Fix a dataset (e.g. CIFAR-10, MNIST)
- Define a *search space A* (e.g., 20 layers of {conv, pool, ReLU, dense})
- Define an *objective function* $f:A\rightarrow[0,1]$
    - $f(a)$ = *validation accuracy of a after training*
- Define a *distance function* $d(a_1, a_2)$ between architectures
    - Quick to evaluate. If $d(a_1, a_2)$ is small, $| f(a_1) - f(a_2) |$ is small

Bayesian optimization

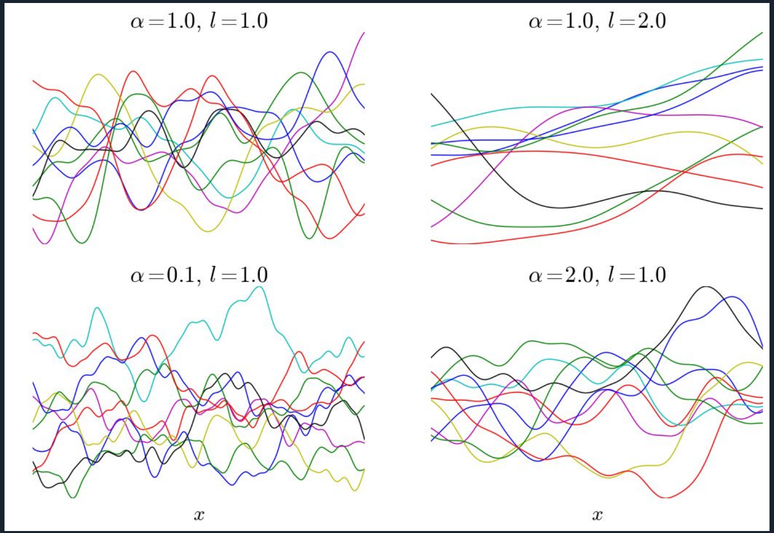
- Fix a dataset (e.g. CIFAR-10, MNIST)
- Define a *search space A* (e.g., 20 layers of {conv, pool, ReLU, dense})
- Define an *objective function* $f{:}A \rightarrow [0,1]$
  - *f(a) = validation accuracy of a after training*
- Define a *distance function* $d(a_1, a_2)$ between architectures
  - Quick to evaluate. If $d(a_1, a_2)$ is small, $| f(a_1) - f(a_2) |$ is small

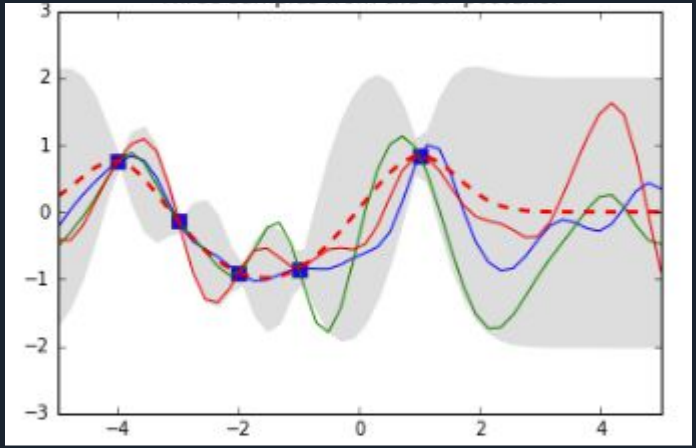
Goal: find $a \in A$ which maximizes $f(a)$

- Choose several random architectures *a* and evaluate $f(a)$
- In each iteration *i*:
  - Use $f(a_1) \ldots f(a_{i-1})$ to choose new a_i
  - Evaluate $f(a_i)$

Gaussian process

- Assume the distribution *f(A)* is *smooth*
- The deviations look like Gaussian noise
- Update as we get more information

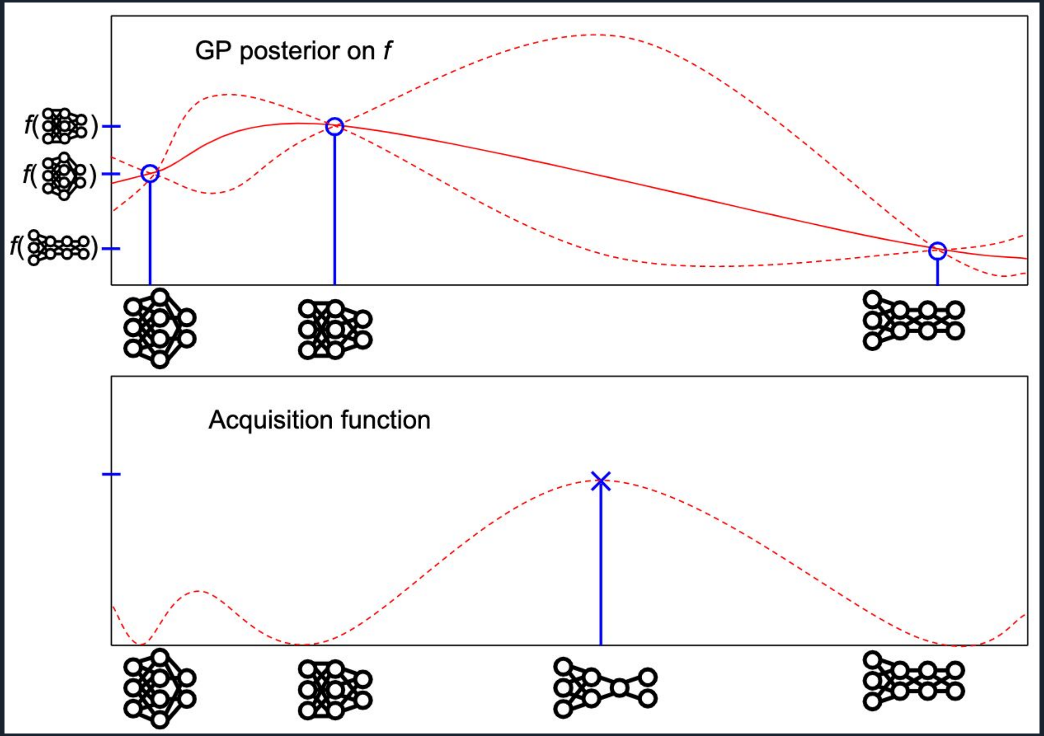




Source: http://keyonvafa.com/gp-tutorial/, https://katbailey.github.io/post/gaussian-processes-for-dummies/

Acquisition function

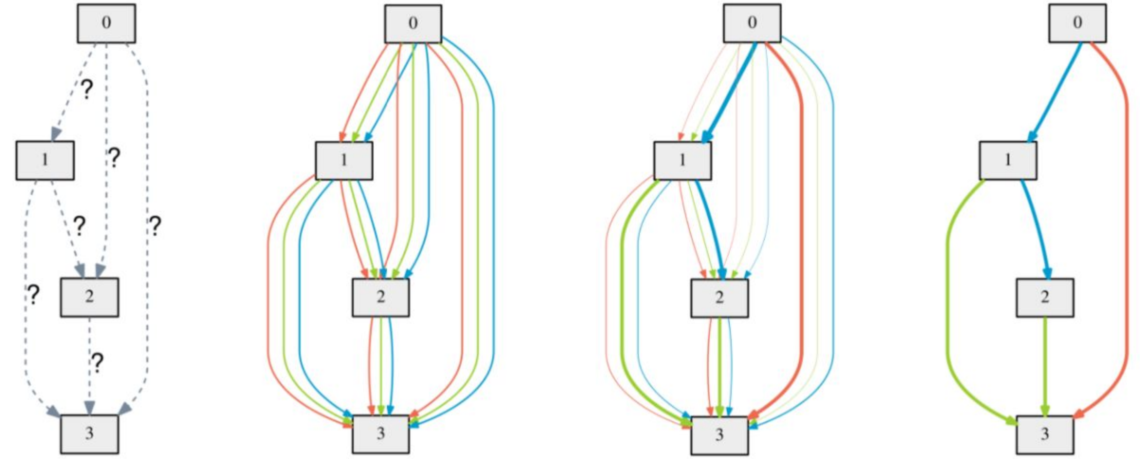
- In each iteration, find the architecture with the largest ***expected improvement***



DARTS: Differentiable Architecture Search

[Liu et al. ‘18]

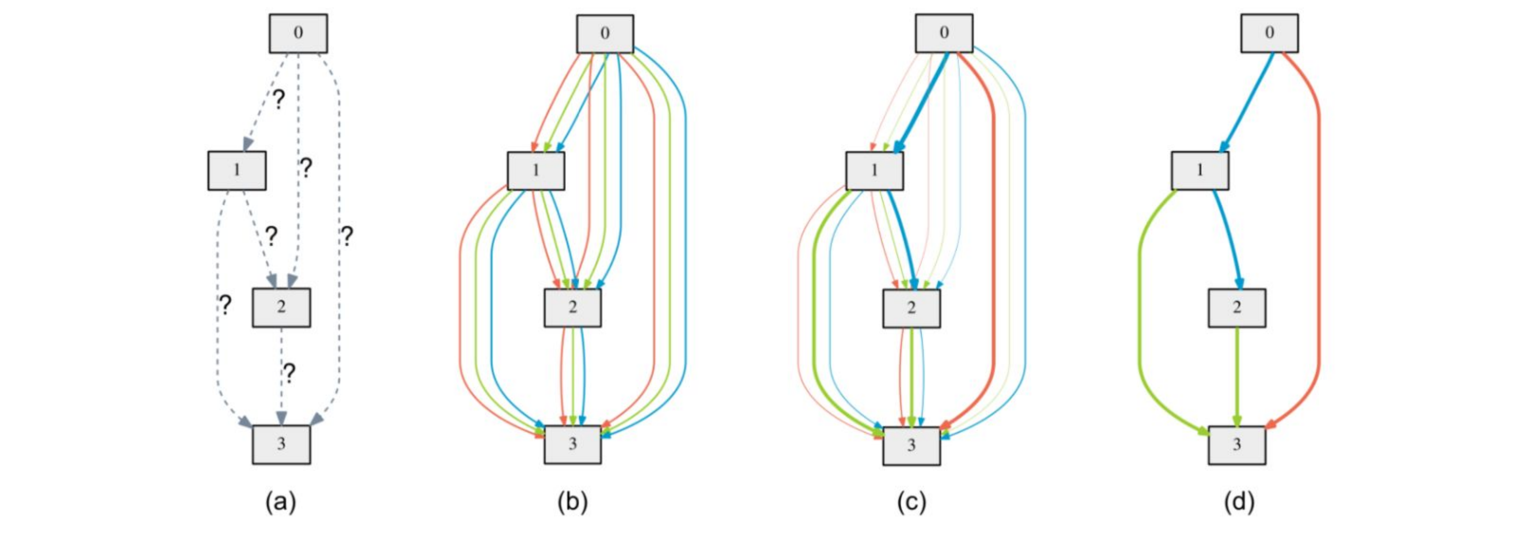
- Relax NAS to a *continuous* problem
- Use gradient descent (just like normal parameters)



(a) (b) (c) (d)

DARTS: Differentiable Architecture Search

[Liu et al. '18]

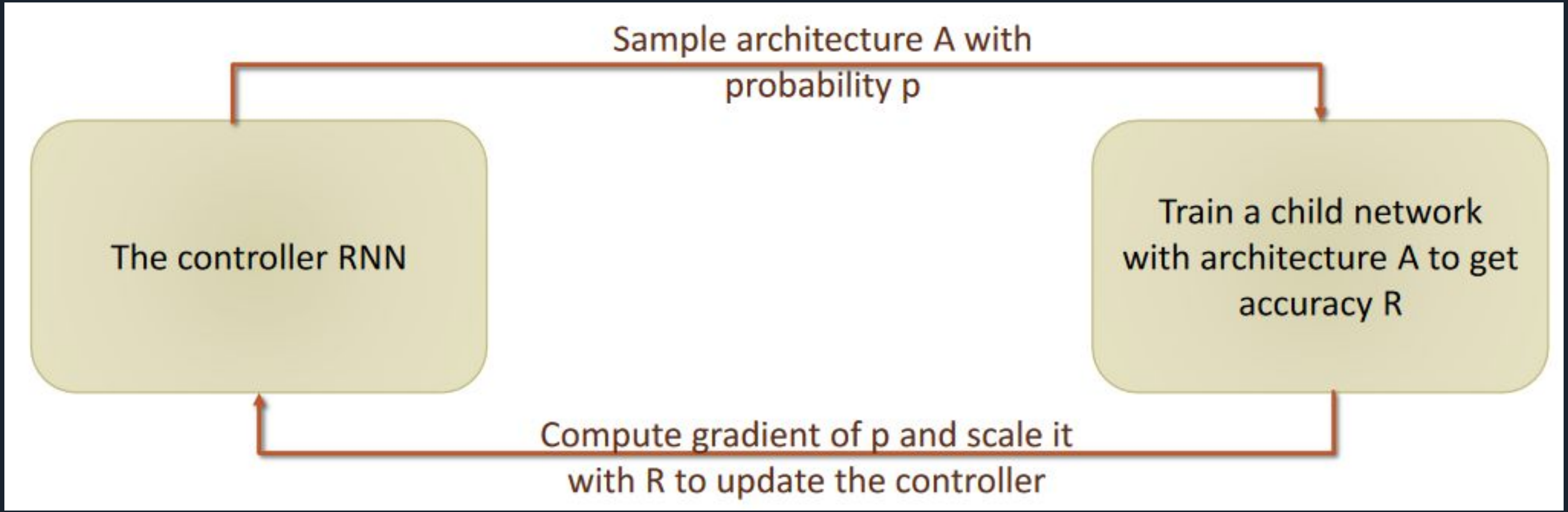


- Upsides: "one-shot"
- Downside: may only work in "micro" search setting

Reinforcement Learning

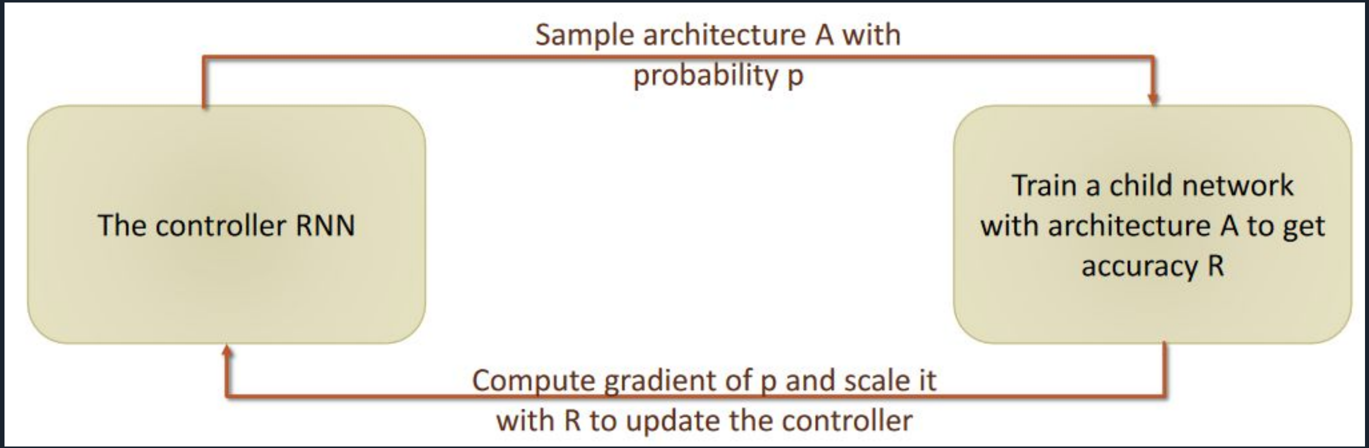
[Zoph, Le ‘16]

- **Controller** recurrent neural network
  - Chooses a new architecture in each round
  - Architecture is trained and evaluated
  - Controller receives feedback



Reinforcement Learning

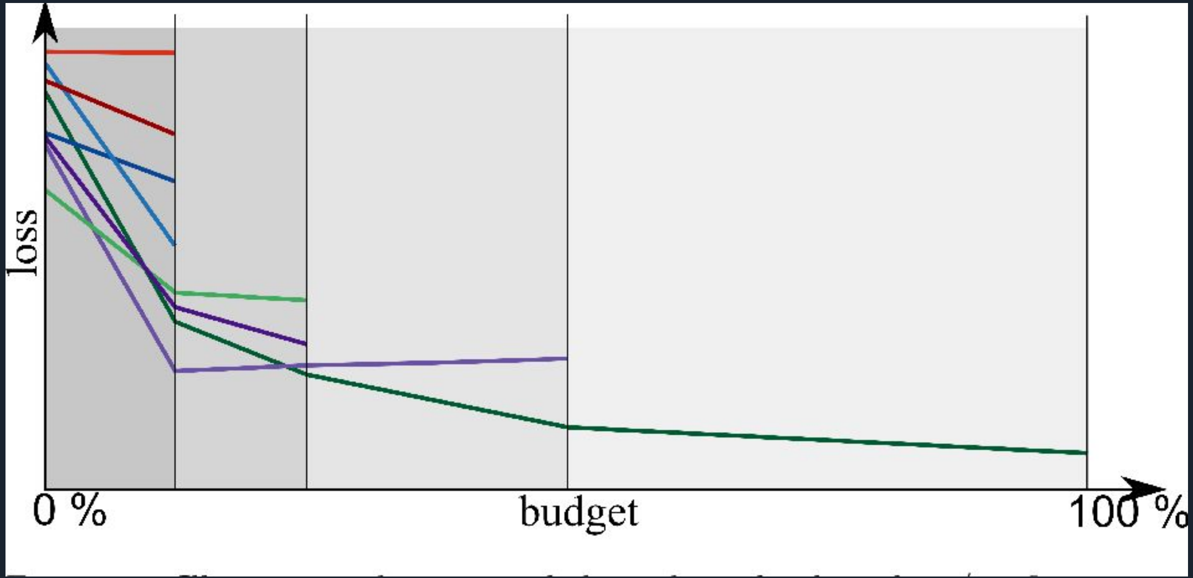
[Zoph, Le ‘16]



- Upside: much more powerful than BayesOpt, gradient descent
- Downsides: train a whole new network using neural networks; RL could be overkill

Evaluation Strategy

- Full training
  - Simple
  - Accurate
- Partial training
  - Less computation
  - Less accurate
- Shared weights
  - Least computation
  - Least accurate



Source:
https://www.automl.org/blog-2nd-automl-challenge/

Is NAS ready for widespread adoption?

- Hard to reproduce results [Li, Talwalkar ‘19]
- Hard to compare different papers
- Search spaces have been getting smaller
- Random search is a strong baseline [Li, Talwalkar ‘19], [Sciuto et al. ‘19]
- Recent papers are giving fair comparisons
- NAS cannot yet consistently beat human engineers

- Auto-Keras tutorial: https://www.pyimagesearch.com/2019/01/07/auto-keras-and-automl-a-getting-started-guide/
- DARTS repo: https://github.com/quark0/darts

Conclusion

- NAS: find the best neural architecture for a given dataset
- Search space, search strategy, evaluation method
- Search strategies: RL, BayesOpt, continuous optimization
- Not yet at the point of widespread adoption in industry

Thanks! Questions?