10-605/10-805: Machine Learning with Large Datasets

Fall 2022

Neural Architecture Search

Announcements

- Guest lecture on Thursday (11/17): Krishna Rangasayee SiMa.ai
  - Topic: ML on Embedded Edge Devices)
  - Virtual / Zoom
  - Will not be recorded — please show up!
  - **Will be covered on Exam 2 HW5 due Wednesday April 6**

WHERE ARE WE NOW?

Key course topics

Data preparation
- Data cleaning
- Data summarization
- Visualization
- Dimensionality reduction

Training
- Distributed ML
- Large-scale optimization
- Scalable deep learning
- Efficient data structures
- Hyperparameter tuning

Inference
- Hardware for ML
- Techniques for low-latency inference (compression, pruning, distillation)

Infrastructure / Frameworks
- Apache Spark
- TensorFlow
- AWS / Google Cloud / Azure

Advanced topics
- Federated learning
- Neural Architecture Search
- Productionizing ML

MOVING FORWARD

What makes deep learning expensive?

Training requires lots of computation

- Specialized hardware (GPUs, TPUs) can help with matrix computations
- Can use advanced iterative optimization methods
- Can parallelize training

Hyperparameter tuning and neural architecture search (NAS) make this worse

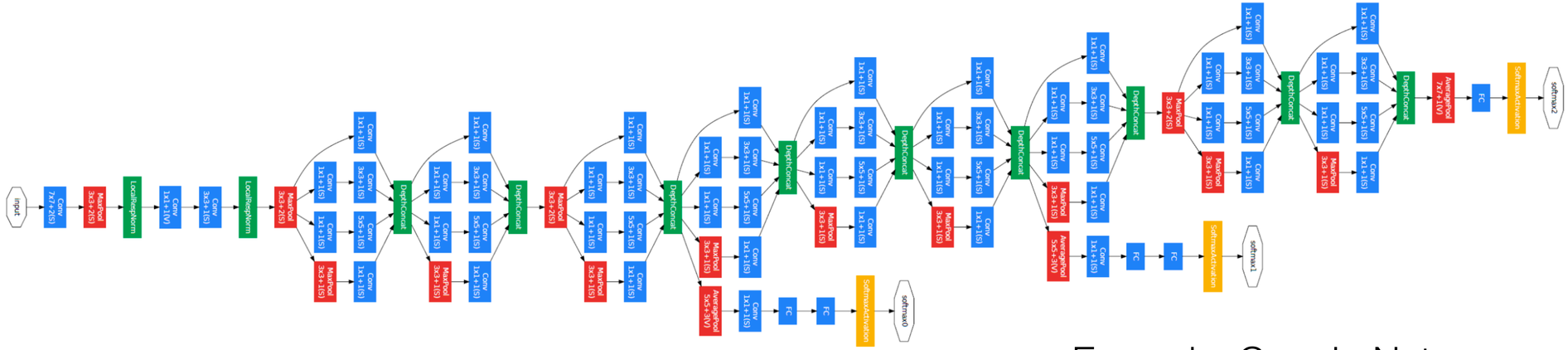
- Lots of knobs to tune!

Resulting models can be large!

- Can be expensive to store model, perform inference

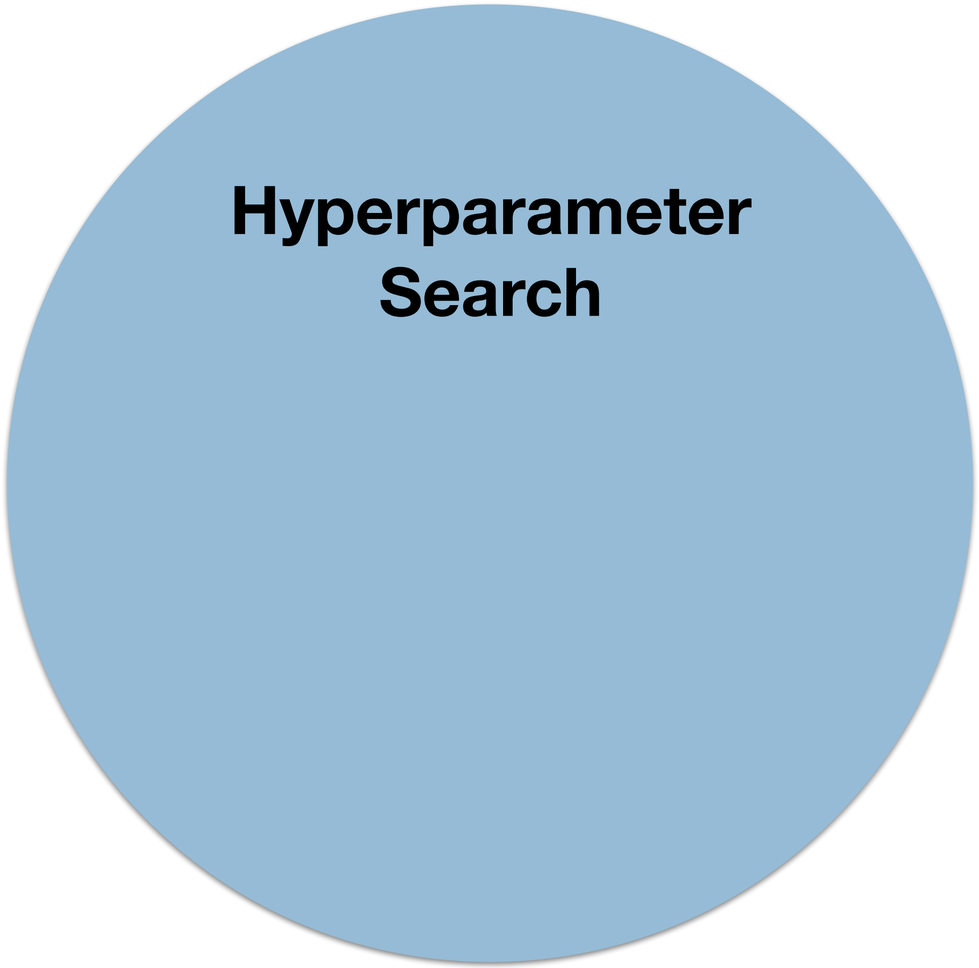
Designing Networks

Ad-hoc, expensive design process restricted to experts!

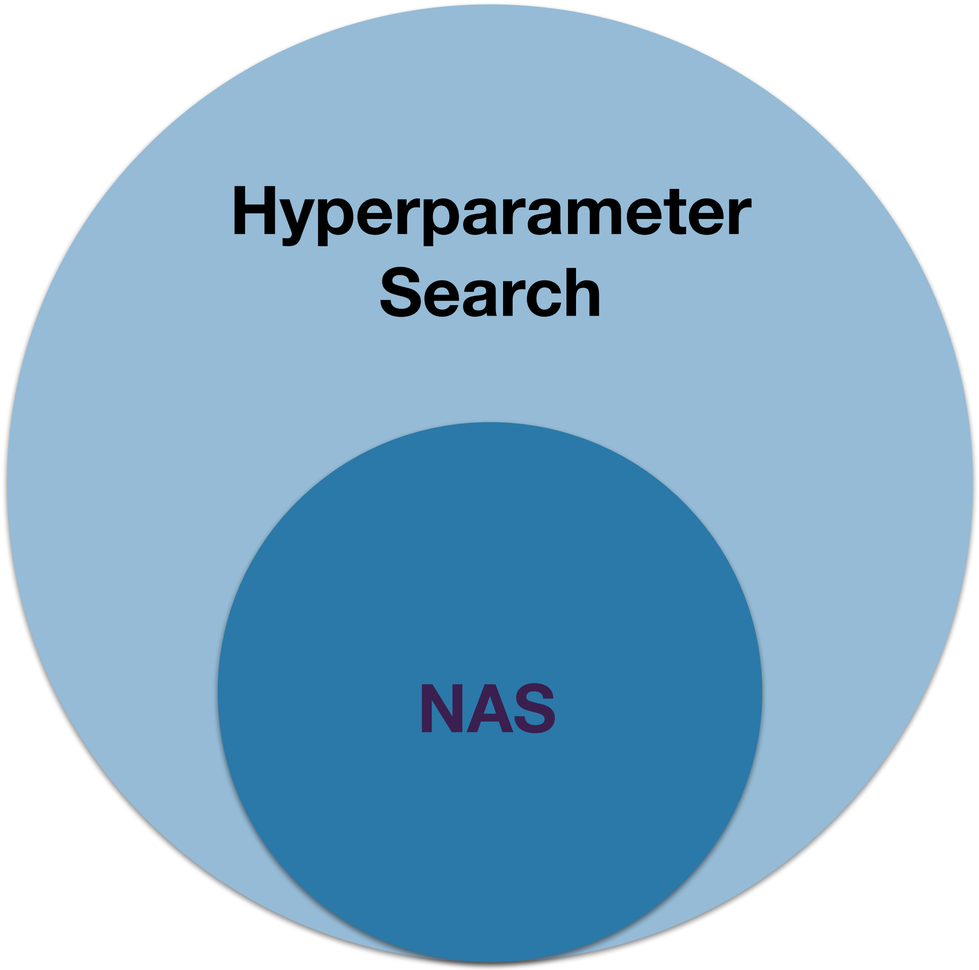


Example: GoogLeNet

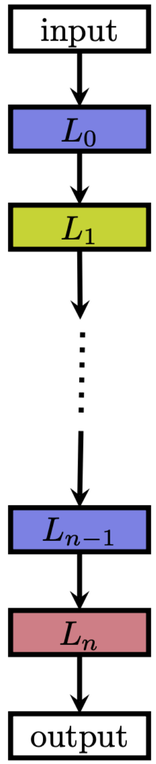
NAS has potential to automate / speed up this process


Hyperparameter
Search


NAS


Hyperparameter
Search
NAS

Hyperparameter Search: Tuning Models

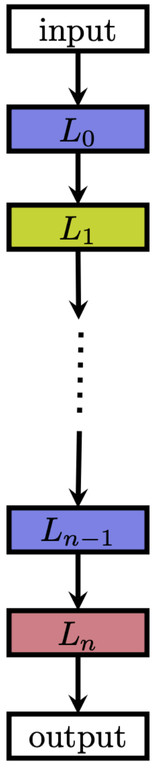


credit: [Elsken, et al., 2019]

Architectural: nodes per layer, number of layers, activation function

Non-architectural: regularization, learning rate, batch-size

Neural Architecture Search: Designing Networks

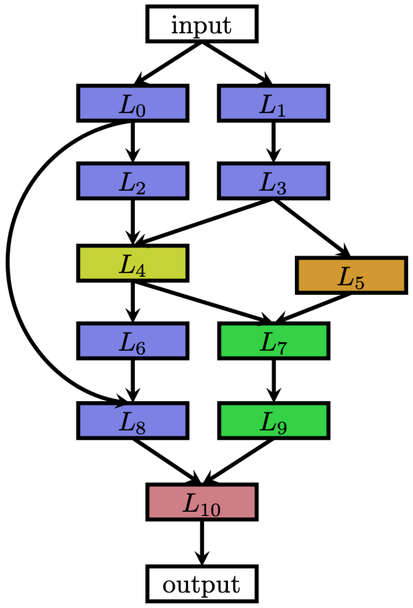


credit: [Elsken, et al., 2019]

Architectural: nodes per layer, number of layers, activation function, *layer operations, network connections*

~~**Non-architectural**: regularization, learning rate, batch-size~~

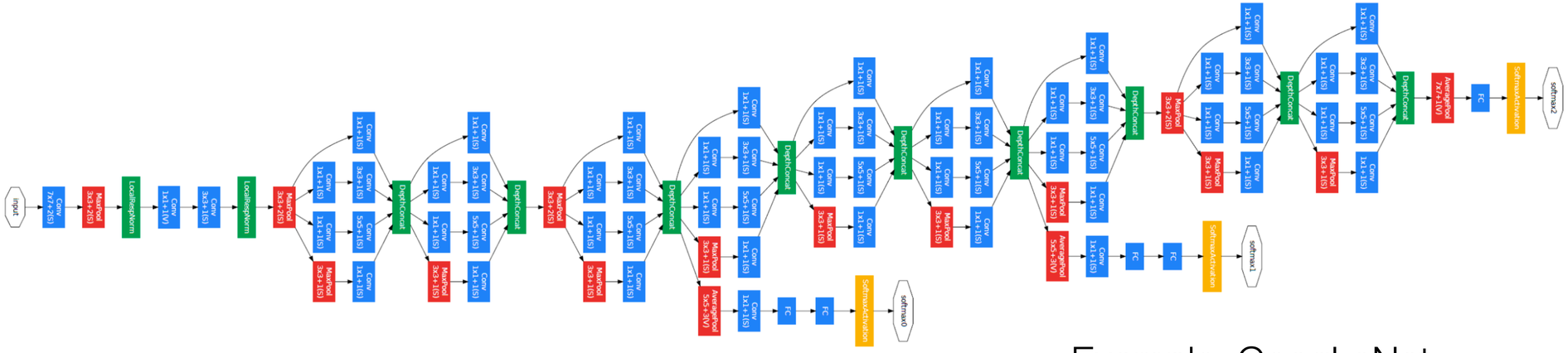
Neural Architecture Search: Designing Networks



credit: [Elsken, et al., 2019]

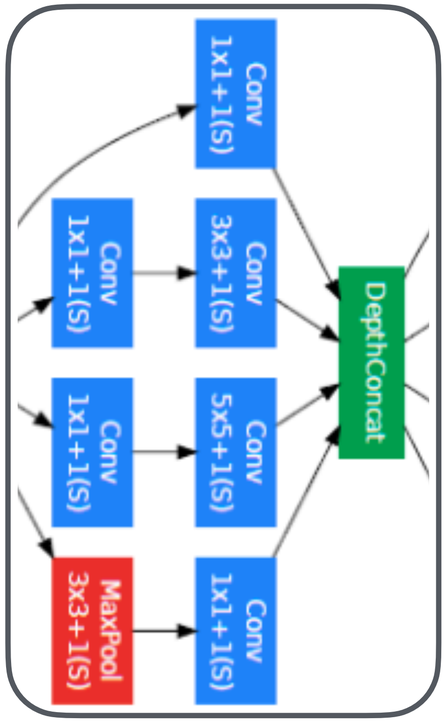
Theory: NAS has potential to replace ad-hoc, expensive design process currently restricted to experts!

Practice: Massive search spaces → intractable problems!



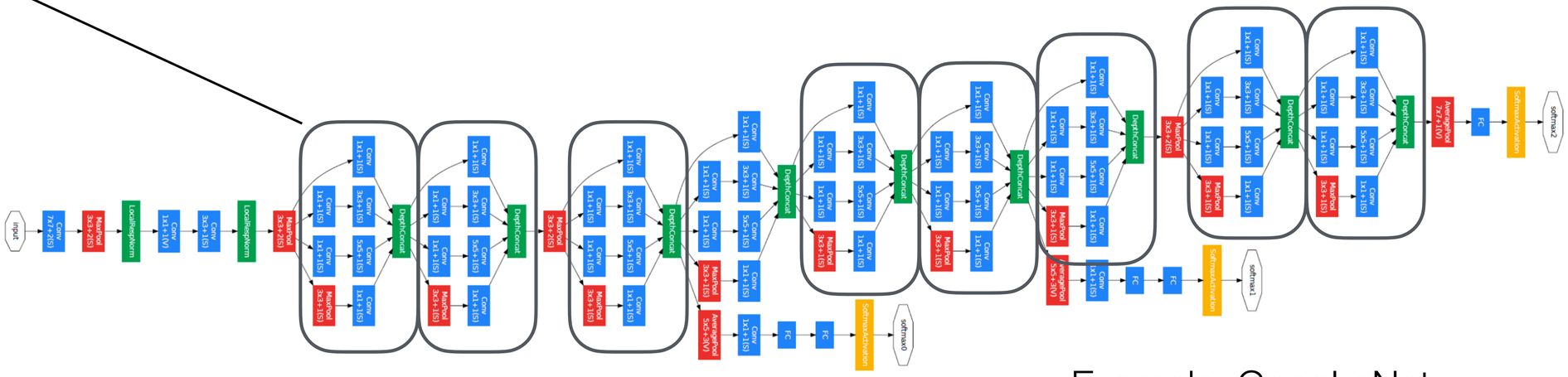
Example: GoogLeNet

Neural Architecture Search: Designing Networks



Inception Module

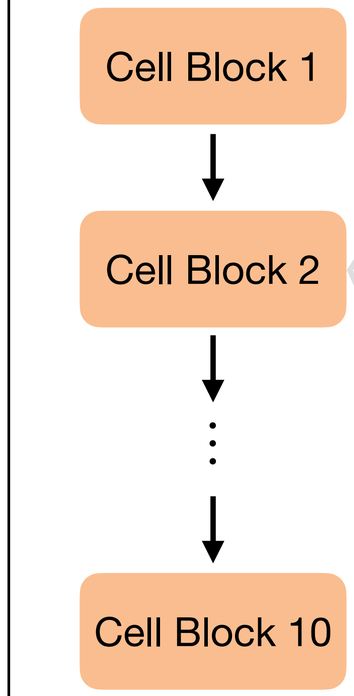
Restrict search to repeated modules or "**cell blocks**"



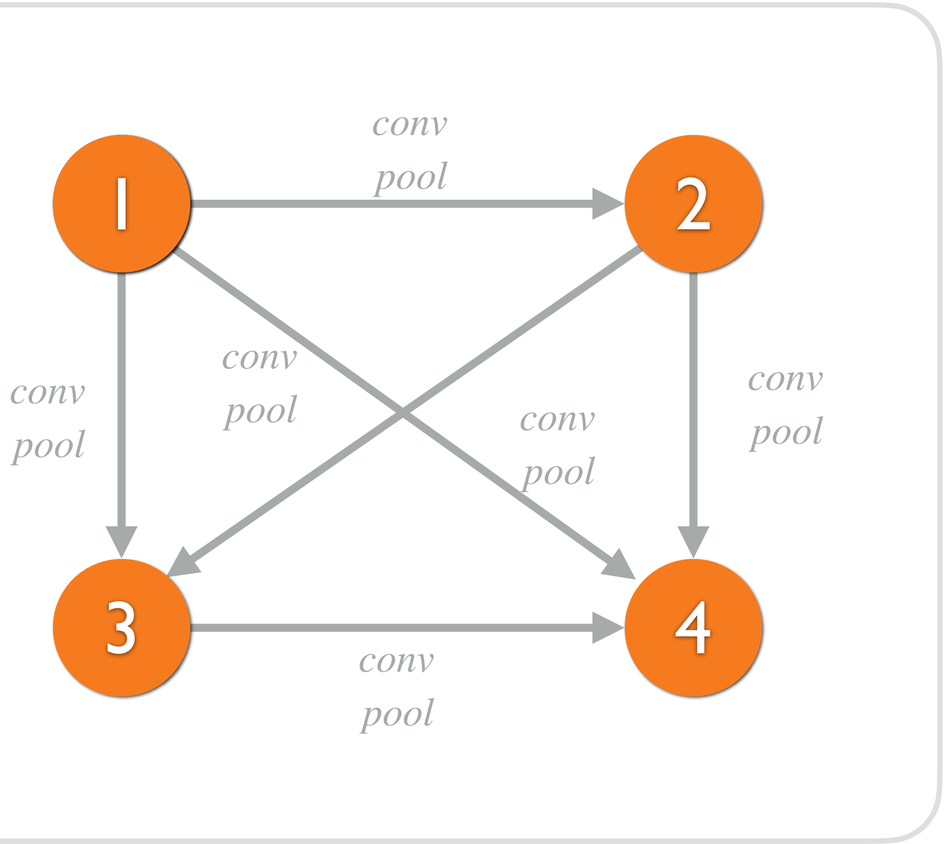
Example: GoogLeNet

Example: Cell Block & Meta Architecture Space

Fixed Meta Architecture

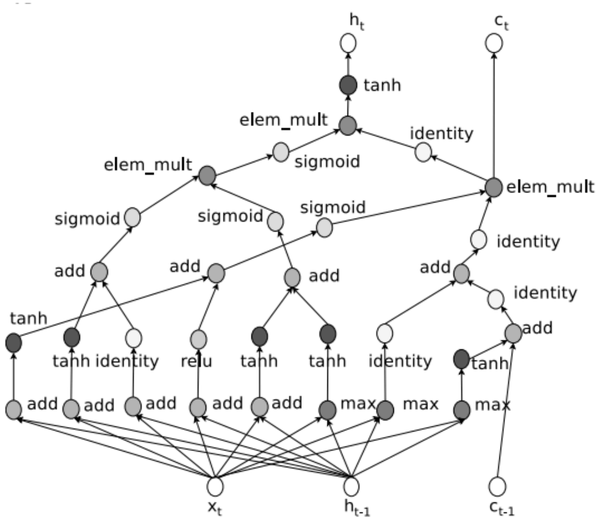


Cell Block Search Space

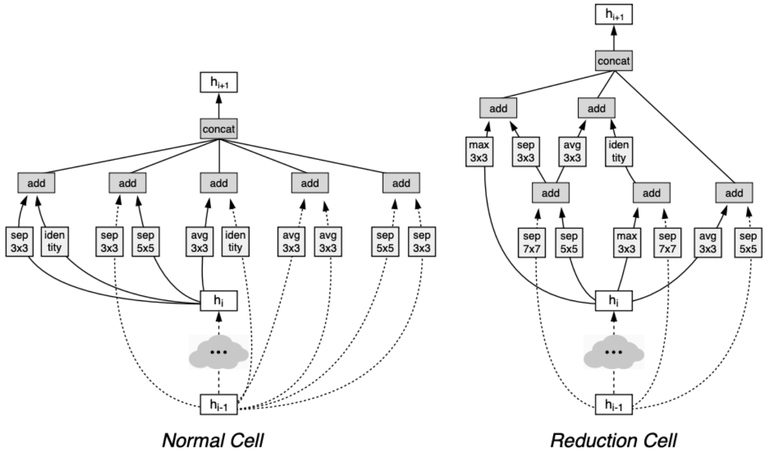


First Generation NAS Methods: state-of-the-art performance…

Penn Treebank



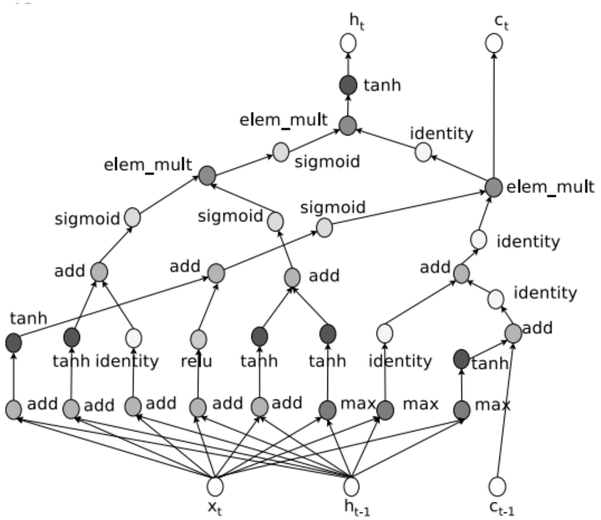
CIFAR-10



Zoph et al., 2016. Reinforcement Learning for NAS.
Zoph et al., 2017. Learning Transferable Architectures for Scalable Image Recognition.

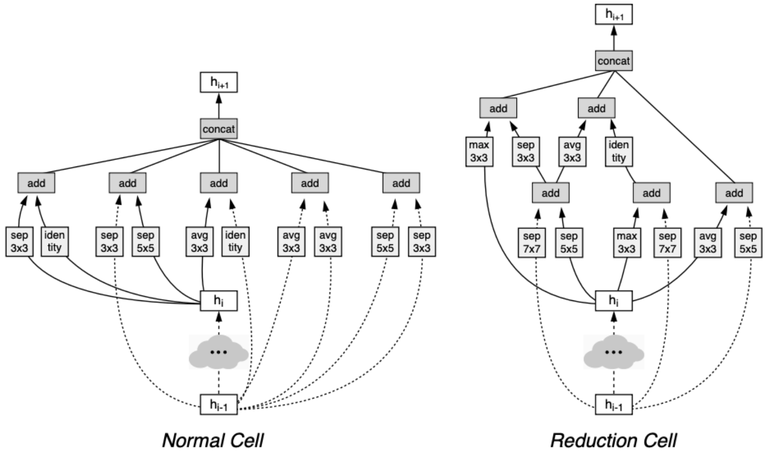
First Generation NAS Methods: state-of-the-art performance… …but incredibly expensive!

Penn Treebank



10k CPU Days!

CIFAR-10



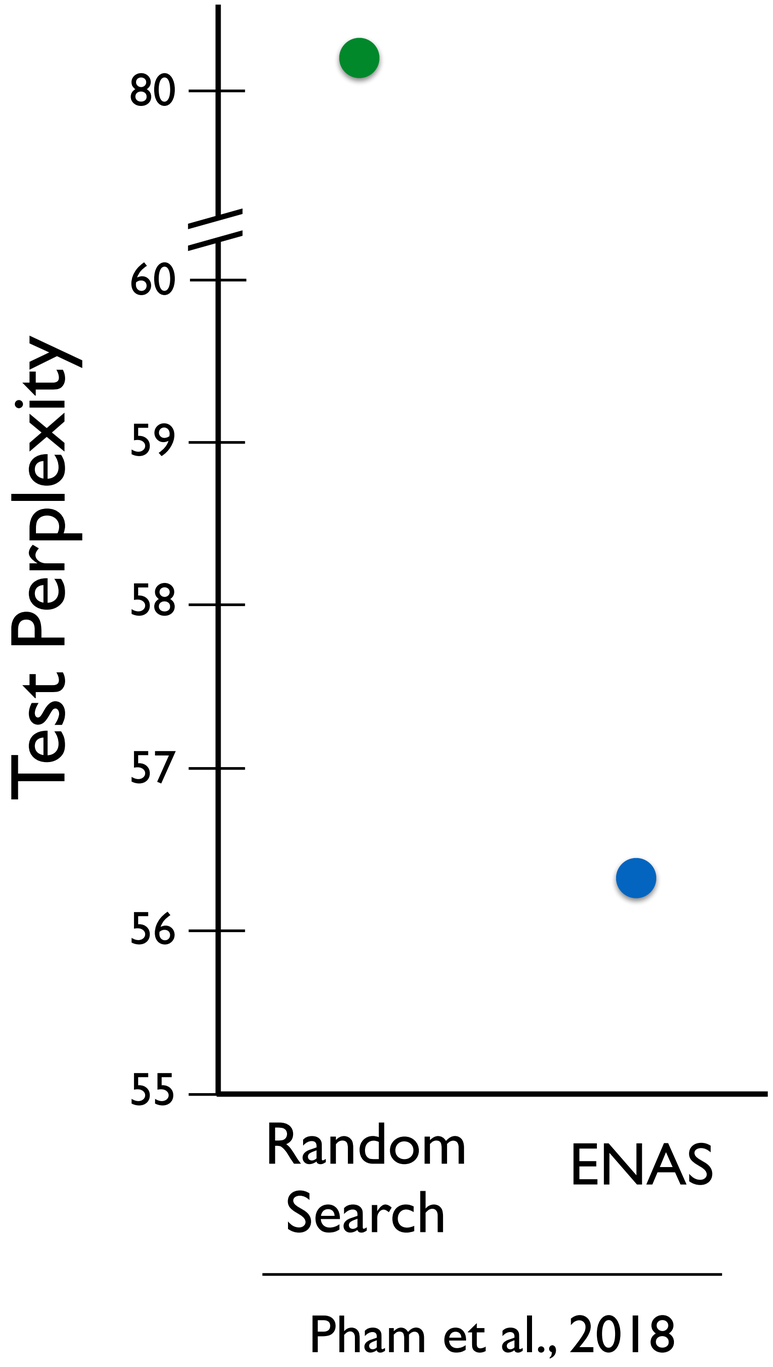
3k GPU Days!

Zoph et al., 2016. Reinforcement Learning for NAS.
Zoph et al., 2017. Learning Transferable Architectures for Scalable Image Recognition.

Soon After: Flurry of New Heuristics Optimizing Two Benchmarks

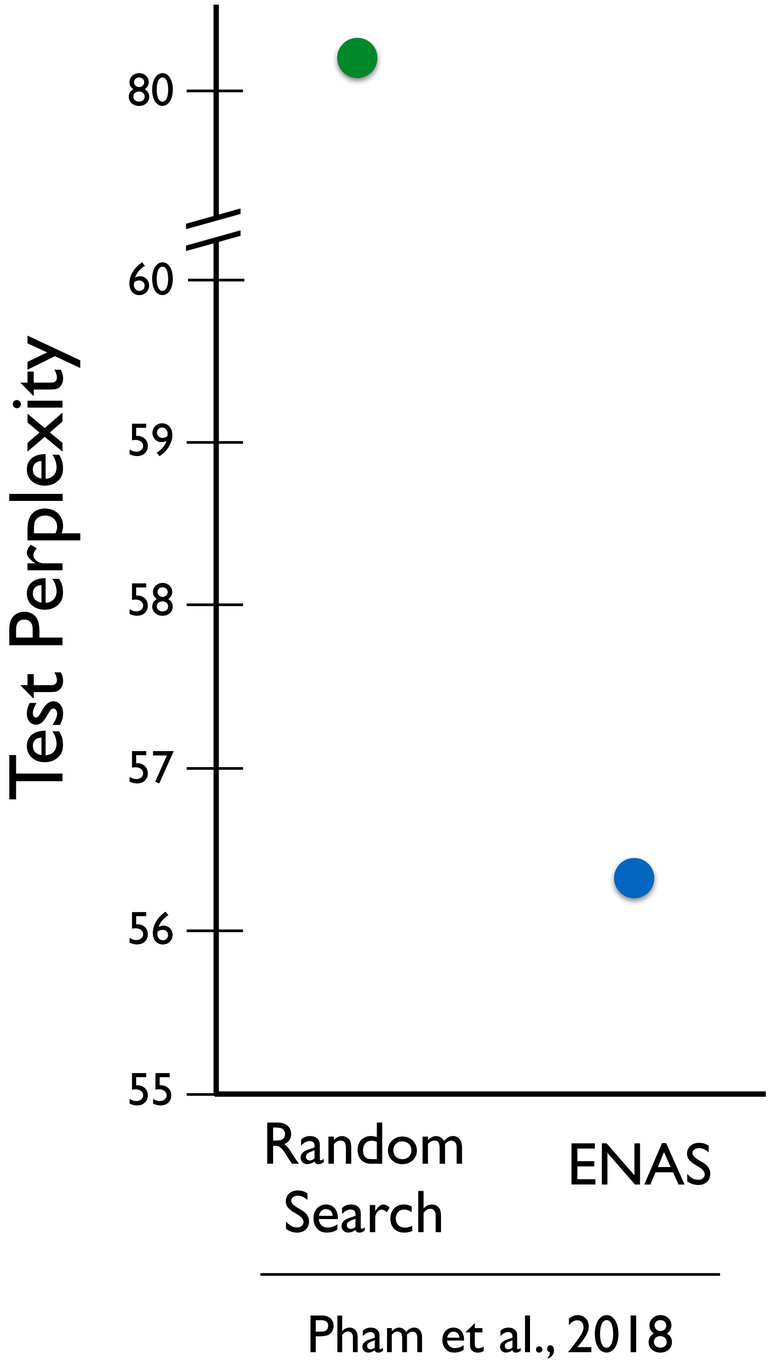
Conference	Authors	Industry Affiliation	Method
ICLR 2018	Brock et al.		SMASH
	Liu et el.	Deepmind	Hierarchical Represntations
ICML 2018	Pham et al.	Google	ENAS
	Cai et al.		Path Level Transformations
	Bender et al.	Google	One - Shot
NIPS 2018	Kandasamy et al.		NASBOT
	Luo et al.	Microsoft	NAO
ICLR 2019	Liu et al.	Deepmind	DARTS
	Cai et al.		Proxyless NAS
	Zhang et al.	Uber	GHN
	Xie et al.	Sensetime	SNAS
	Cao et al.		Architecture Embeddings

SOTA perplexity



Pham et al., 2018

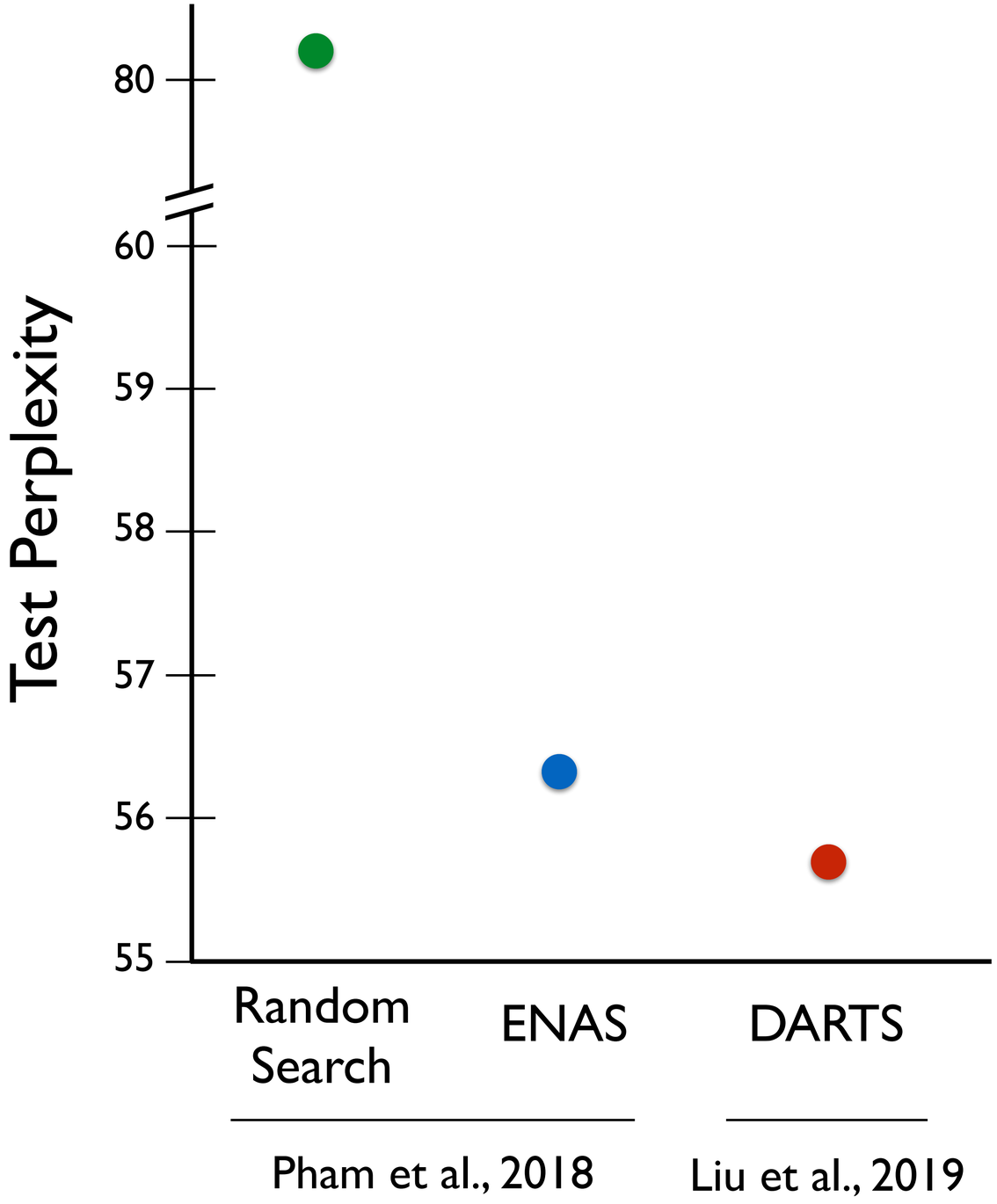
Penn Treebank Benchmark



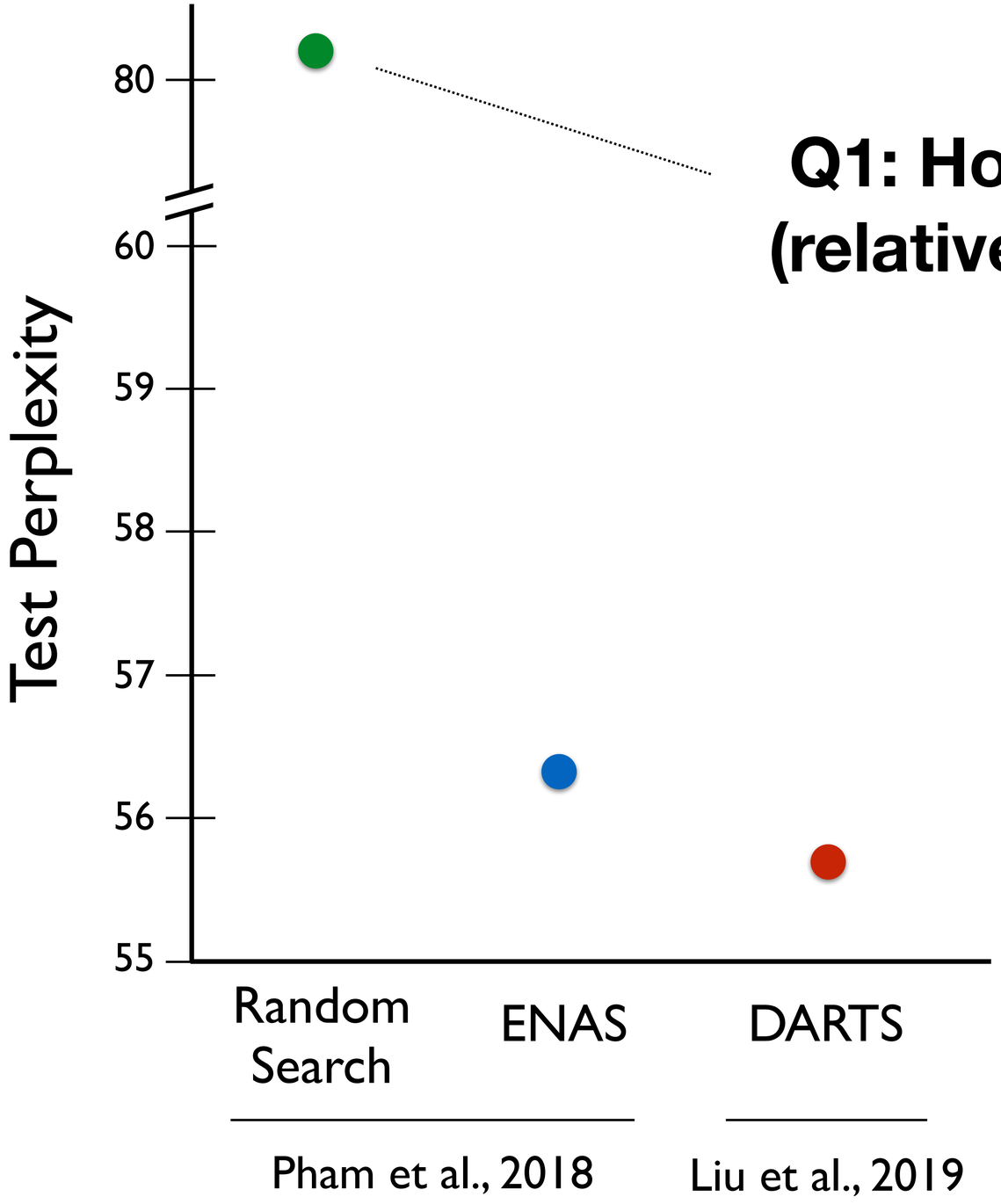
SOTA perplexity and three(!) orders of magnitude faster

Penn Treebank Benchmark

SOTA perplexity, again



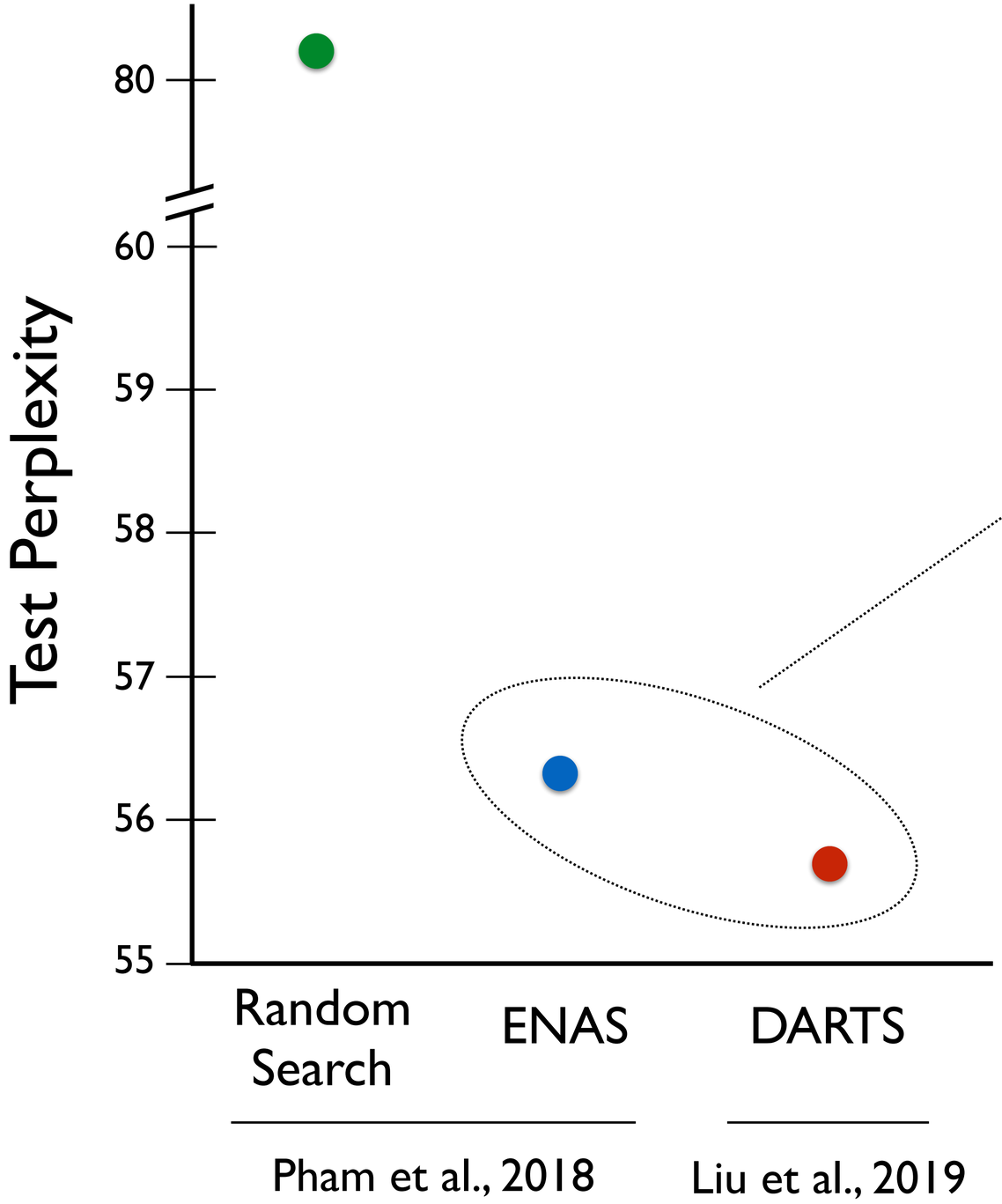
Penn Treebank Benchmark



Q1: How good are NAS heuristics (relative to appropriate baselines)?

Penn Treebank Benchmark

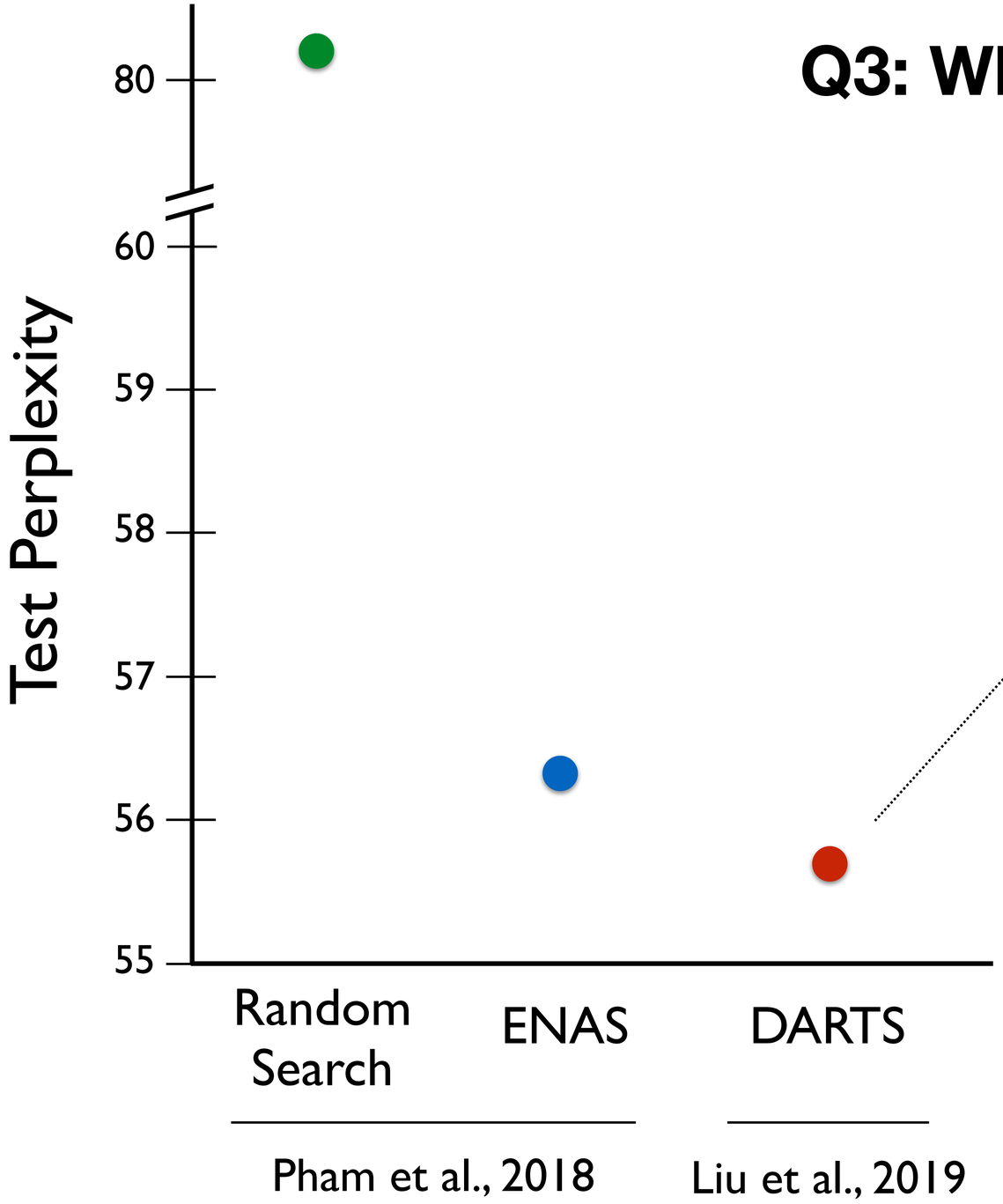
Q2: What algorithmic components matter?



Conference	Authors	Industry Affiliation	Method
ICLR 2018	Brock et al.		SMASH
	Liu et al.	Deepmind	Hierarchical Representations
ICML 2018	Pham et al.	Google	ENAS
	Cai et al.		Path Level Transformations
	Bender et al.	Google	One-Shot
NIPS 2018	Kandasamy et al.		NASBOT
	Luo et al.	Microsoft	NAO
ICLR 2019	Liu et al.	Deepmind	DARTS
	Cai et al.		ProxylessNAS
	Zhang et al.	Uber	GHN
	Xie et al.	Sensetime	SNAS
	Cao et al.		Architecture Embeddings

Penn Treebank Benchmark

Q3: What about problems beyond NLP / CV?



Penn Treebank Benchmark

Q1: How good are NAS heuristics?

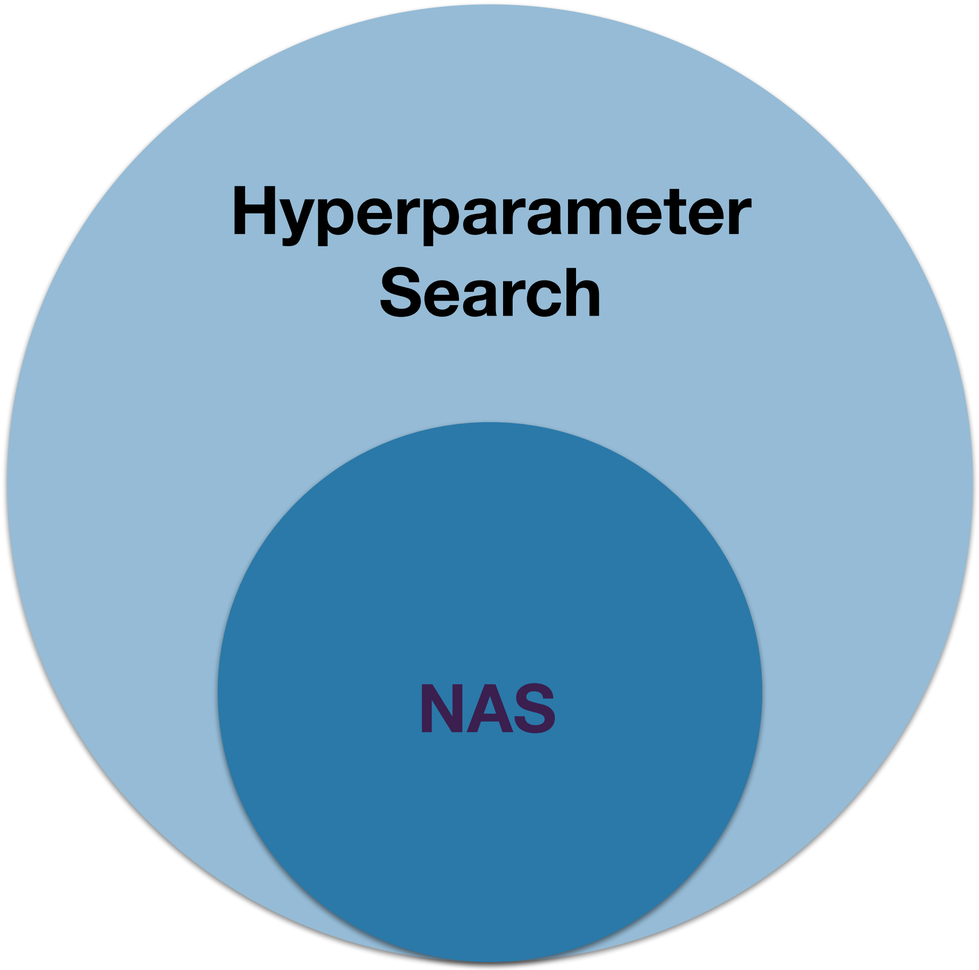
Q2: What algorithmic components matter?

Q3: What about problems beyond NLP / CV?

ICLR17, JMLR18: L. Li (CMU), K. Jamieson (Washington), A. Rostamizadeh (Google), G. Desalvo (Google), A. Talwalkar (CMU, Determined AI)

UAI19: L. Li (CMU), A. Talwalkar (CMU, Determined AI)

MLSys20: L. Li (CMU), K. Jamieson (Washington), A. Rostamizadeh (Google), K. Gonina (Google), J. Ben-tzur (Determined AI), M. Hardt (Berkeley), B. Recht (Berkeley), A. Talwalkar (CMU, Determined AI)



Apply SOTA Hyperparameter Search methods to NAS

Search Space

Continuous
& Discrete

activation fct

nodes per layer

hidden layers

regularization

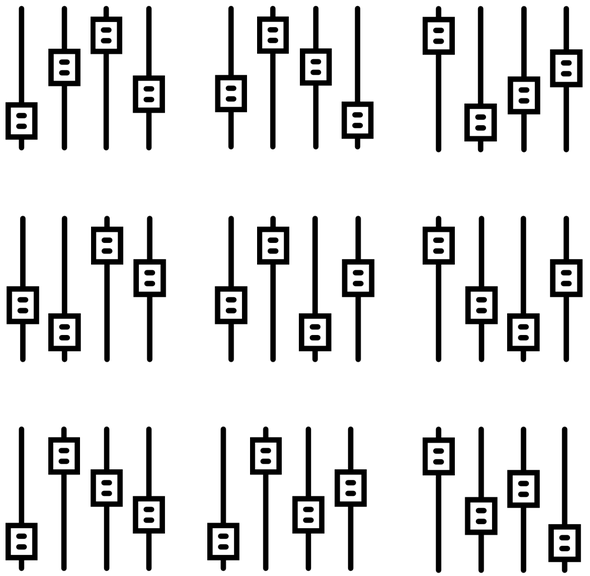
Search Space

Continuous
& Discrete

Search Method

Random Search

activation fct
nodes per layer
hidden layers
regularization



Search Space

Continuous
& Discrete

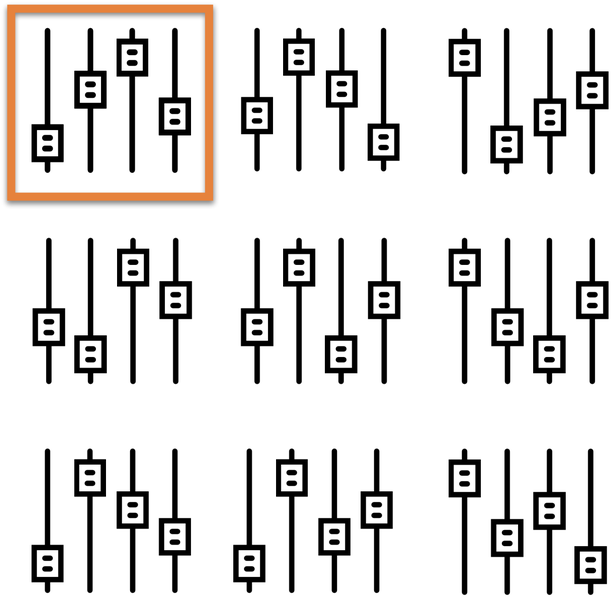
Search Method

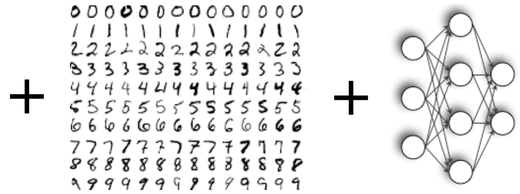
Random Search

Evaluation Method

Full Training

activation fct
nodes per layer
hidden layers
regularization





Black-box Solver / Validator

Error: `0.058`

Search Space

Continuous
& Discrete

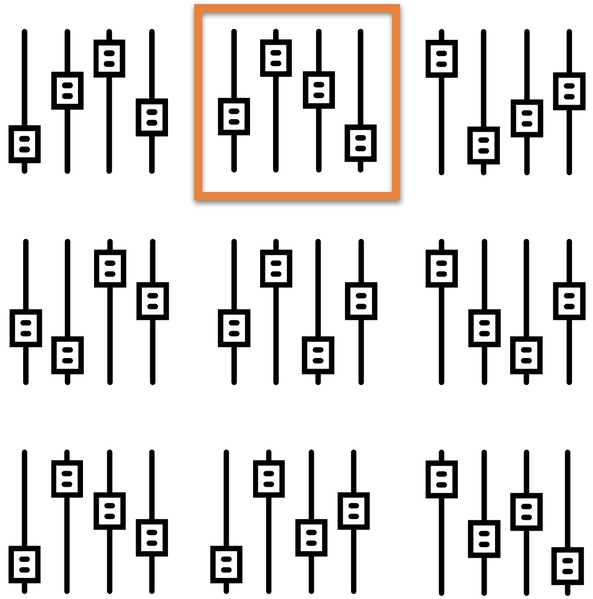
Search Method

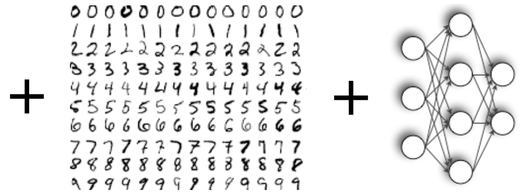
Random Search

Evaluation Method

Full Training

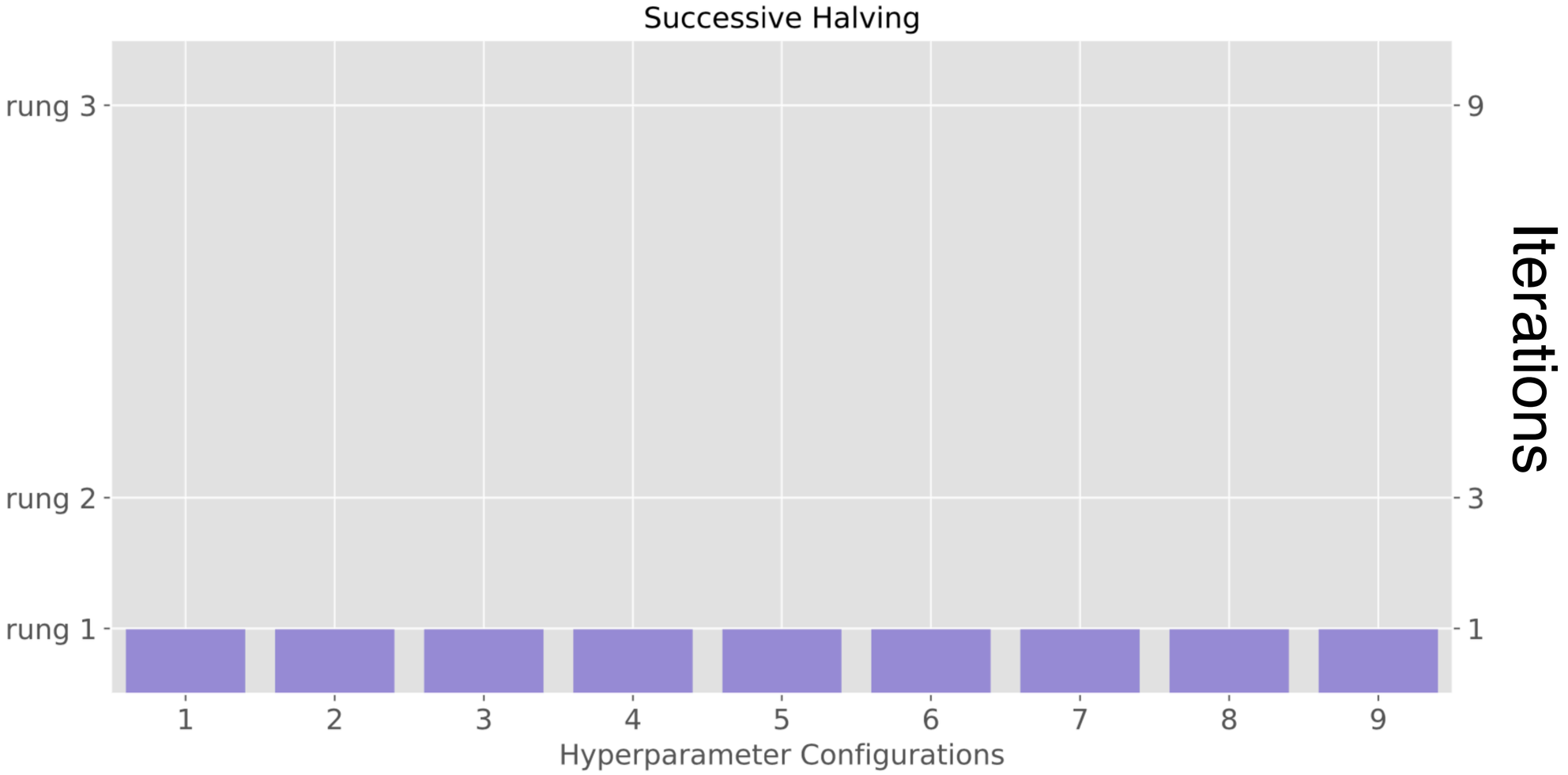
activation fct
nodes per layer
hidden layers
regularization

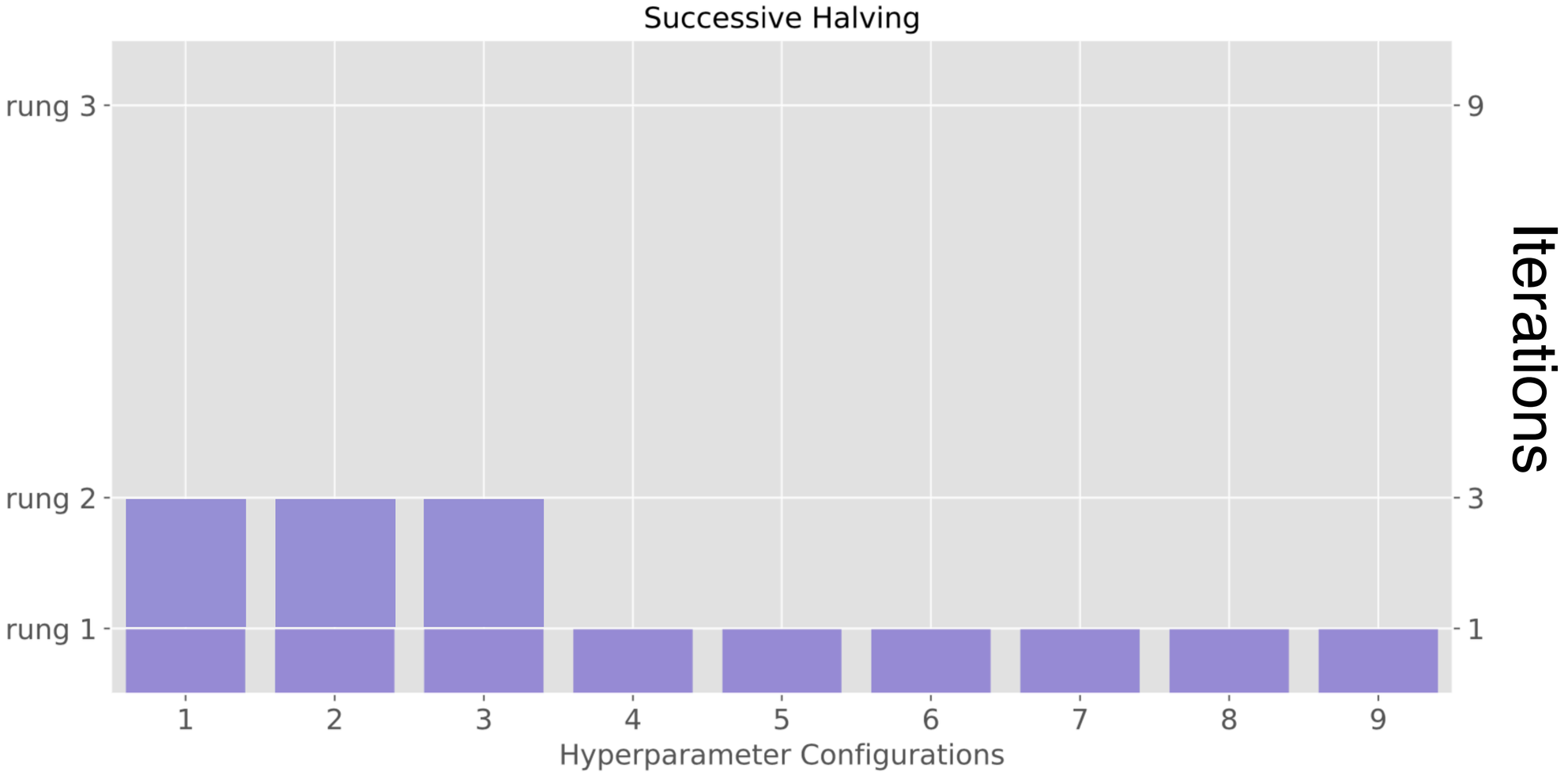


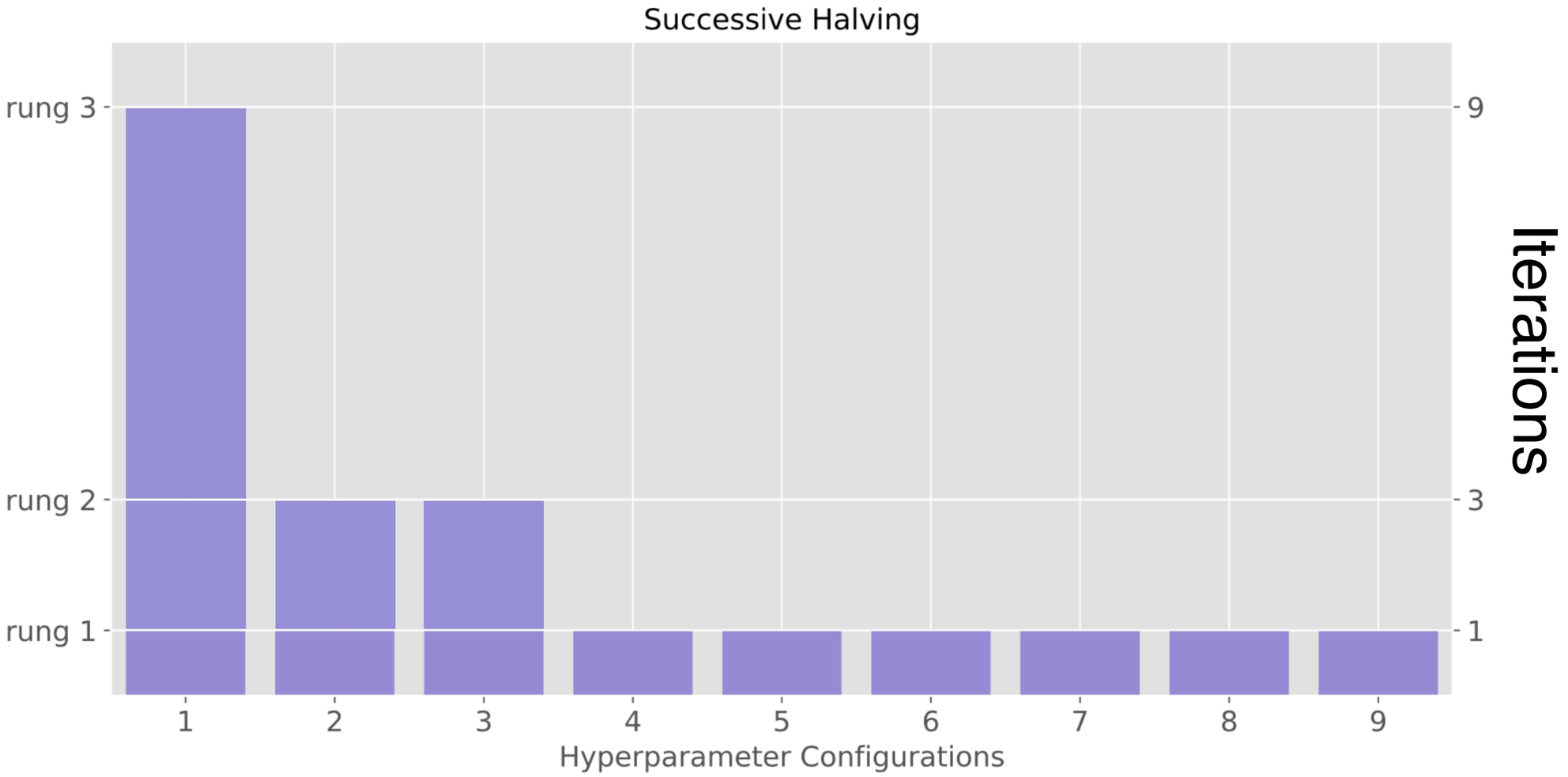


Black-box Solver / Validator

Error: `0.052`


Successive Halving
rung 3
rung 2
rung 1
9
3
1
Iterations
1
2
3
4
5
6
7
8
9
Hyperparameter Configurations


Successive Halving
rung 3
rung 2
rung 1
9
3
1
Iterations
1
2
3
4
5
6
7
8
9
Hyperparameter Configurations


Successive Halving
rung 3
rung 2
rung 1
9
3
1
Iterations
1
2
3
4
5
6
7
8
9
Hyperparameter Configurations

Hyperband / ASHA

Novel **downsampling** approach

[AISTATS16, ICLR17, JMLR18, MLSys20]

✔ **State-of-the-art** empirical performance

✔ **Provably correct**

Theory: Frame as a **multi-armed bandit** problem

Practice: Asynchronous execution for **massive parallelism**

Large-scale Regime of HP Optimization

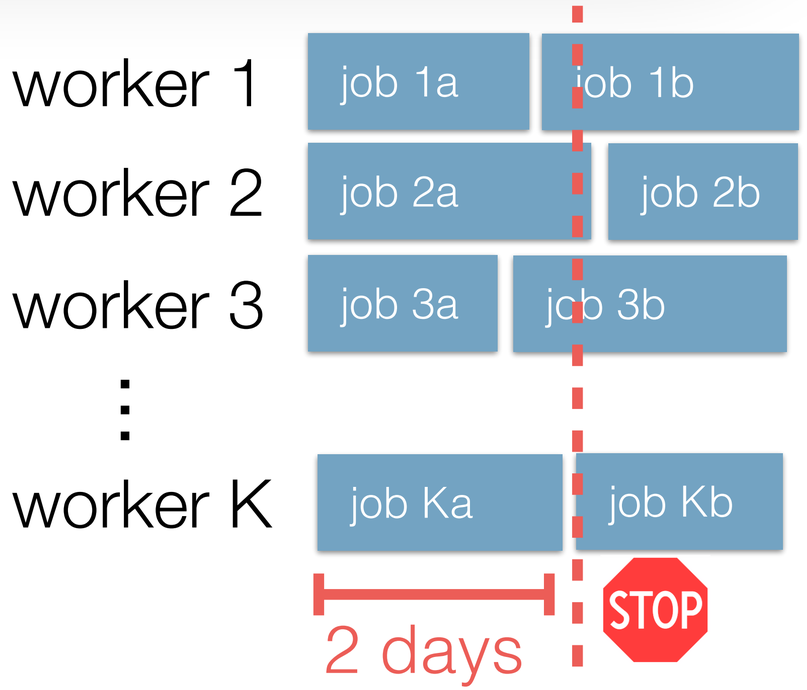
Goal: Evaluate # models » # workers

Budget: Time needed to train a few models

High dimensional settings

Must consider many models relative to # workers

(e.g., 10K models vs 100 GPUs)



Large-scale Regime of HP Optimization

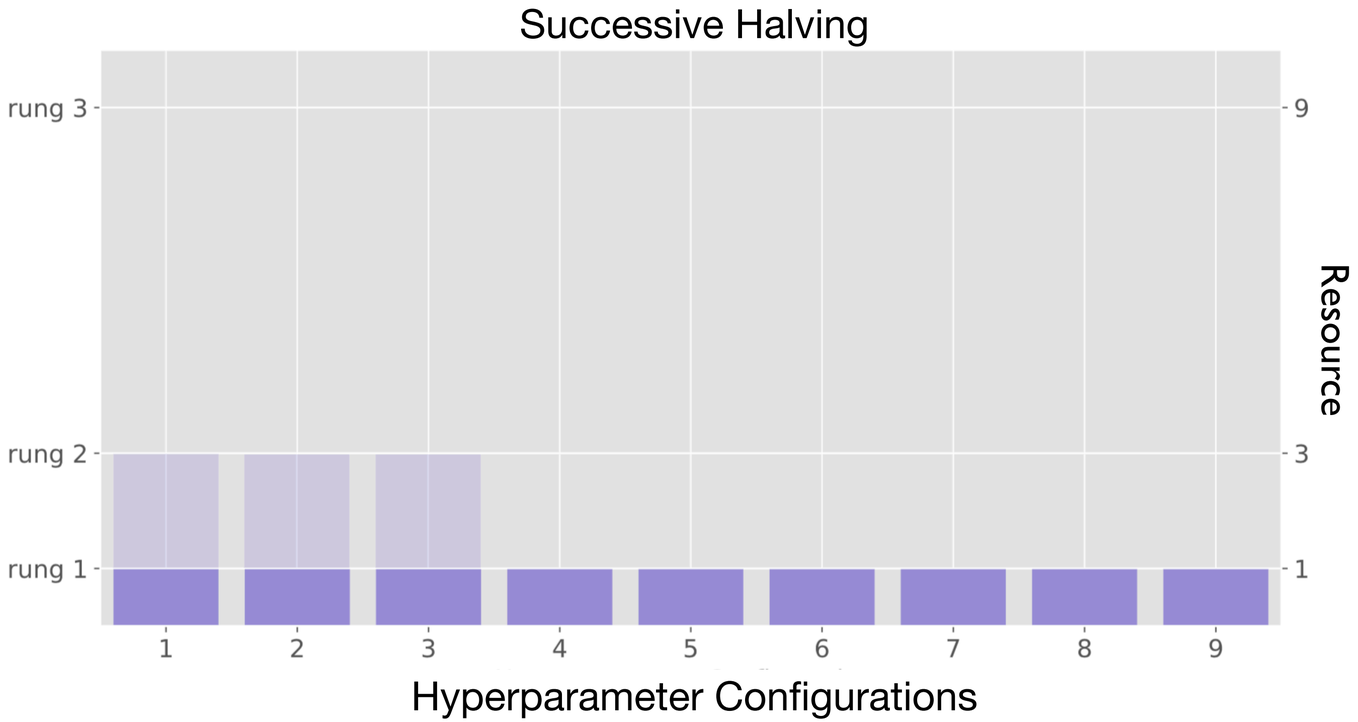
Goal: Evaluate # models » # workers

Budget: Time needed to train a few models

- ✗ **Not enough workers** to parallelize random search
- ✗ High dimensional spaces and parallelism **limit adaptivity**

ASHA: Asynchronous, massively-parallel generalization of Hyperband *[MLSys20]*

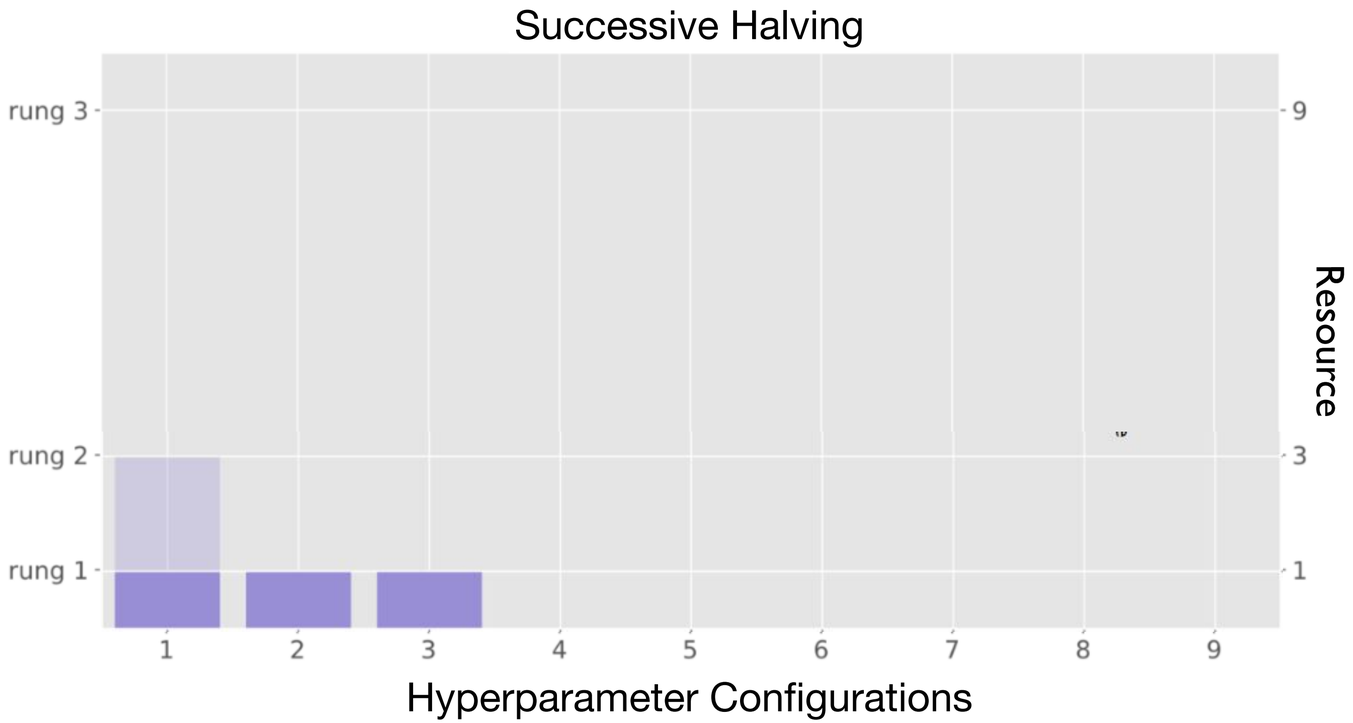
Naively Parallelizing Successive Halving



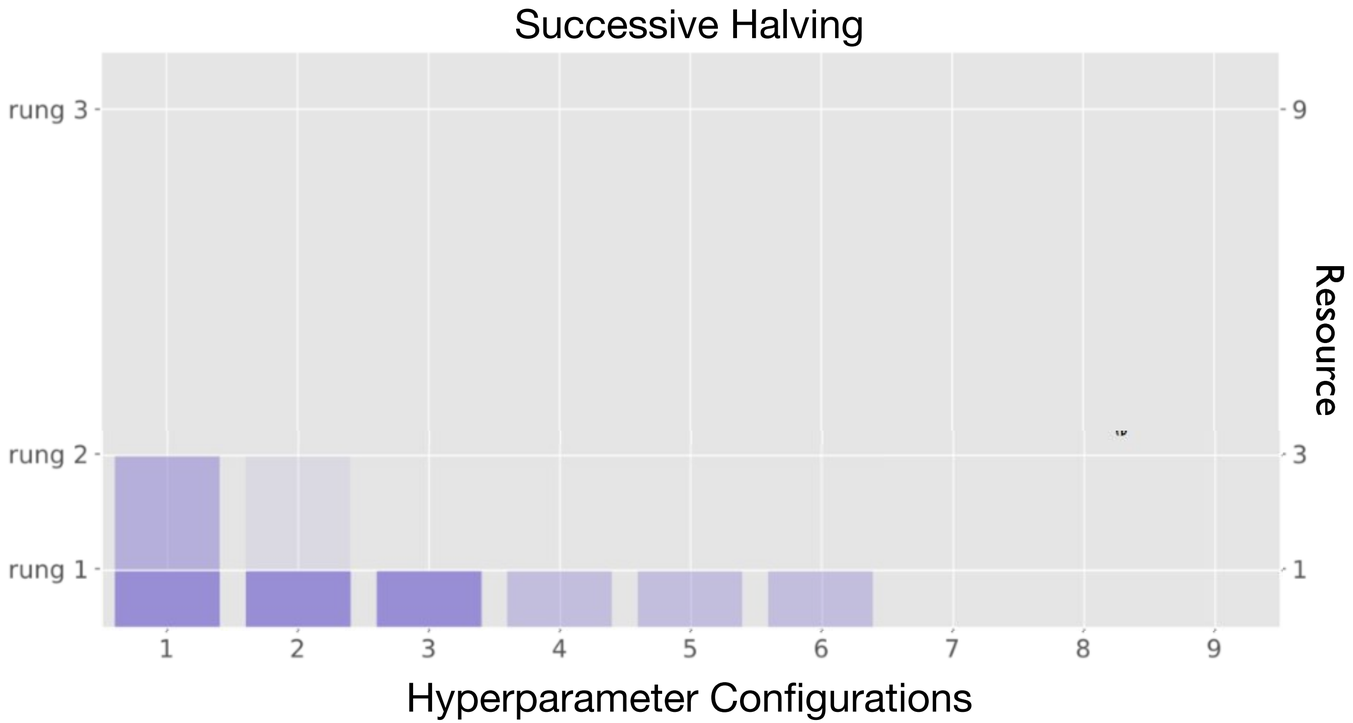
Synchronous promotions hinder utilization

Compounded when dealing with **stragglers** / **pre-emption**

Key Idea: Promote when possible!



Key Idea: Promote when possible!



Asynchrony improves utilization

Robust to stragglers / pre-emption

Promotes a **vanishing fraction of erroneous configurations**


Microsoft


Preferred
Networks


OPTUNA


facebook


Neural Network Intelligence


TensorFlow


Katib


PYTORCH


scikit
learn


DASK


Determined AI


Keras


RAY


mxnet

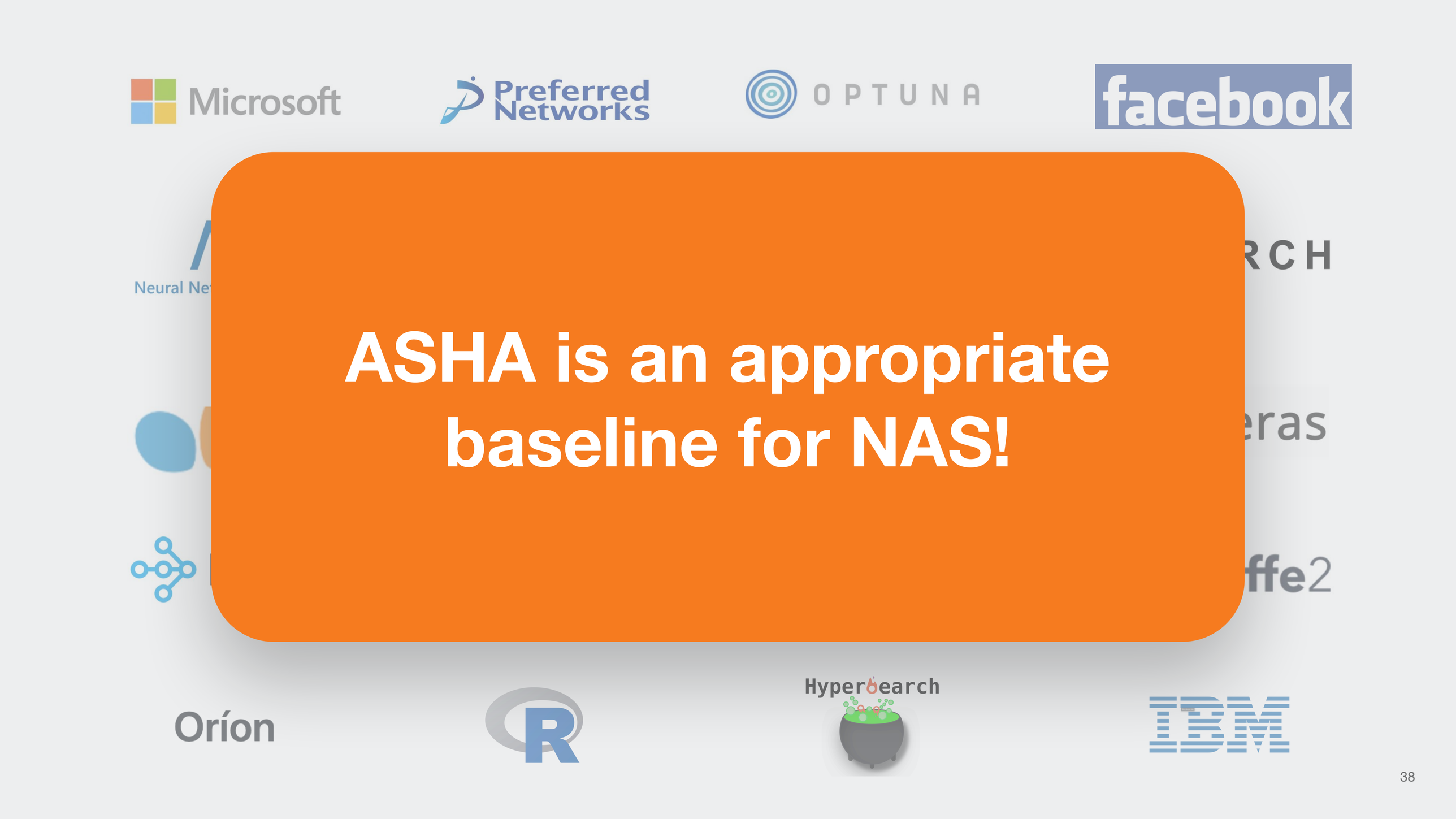

Polyaxon


Caffe2


Oríon


Hyperbearch

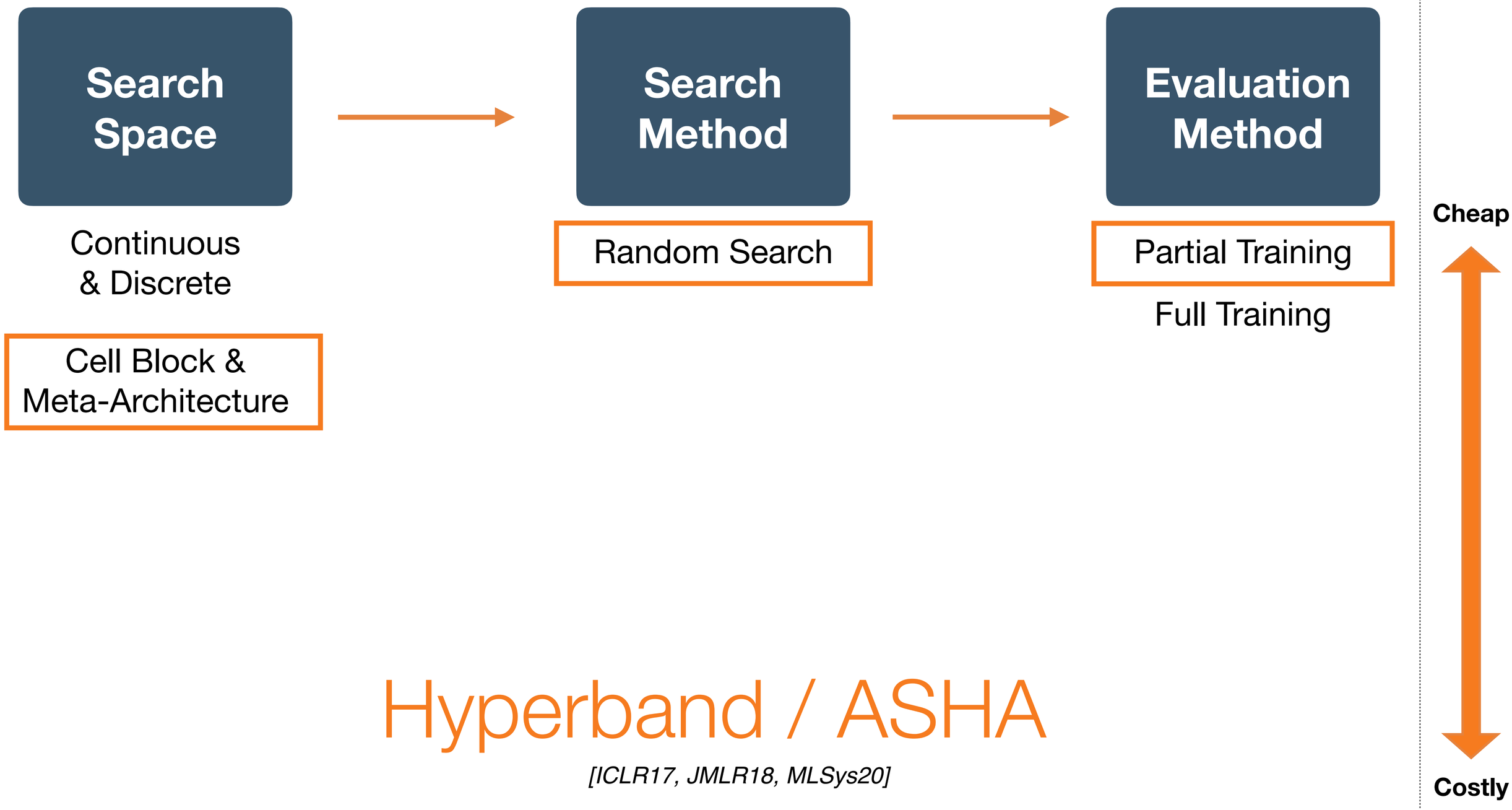

IBM


Microsoft
Preferred Networks
OPTUNA
facebook
RCH
eras
ffe2
Oríon
R
HyperOearch
IBM
ASHA is an appropriate baseline for NAS!


Search Space
Search Method
Evaluation Method
Continuous & Discrete
Random Search
Full Training
Cheap
Costly

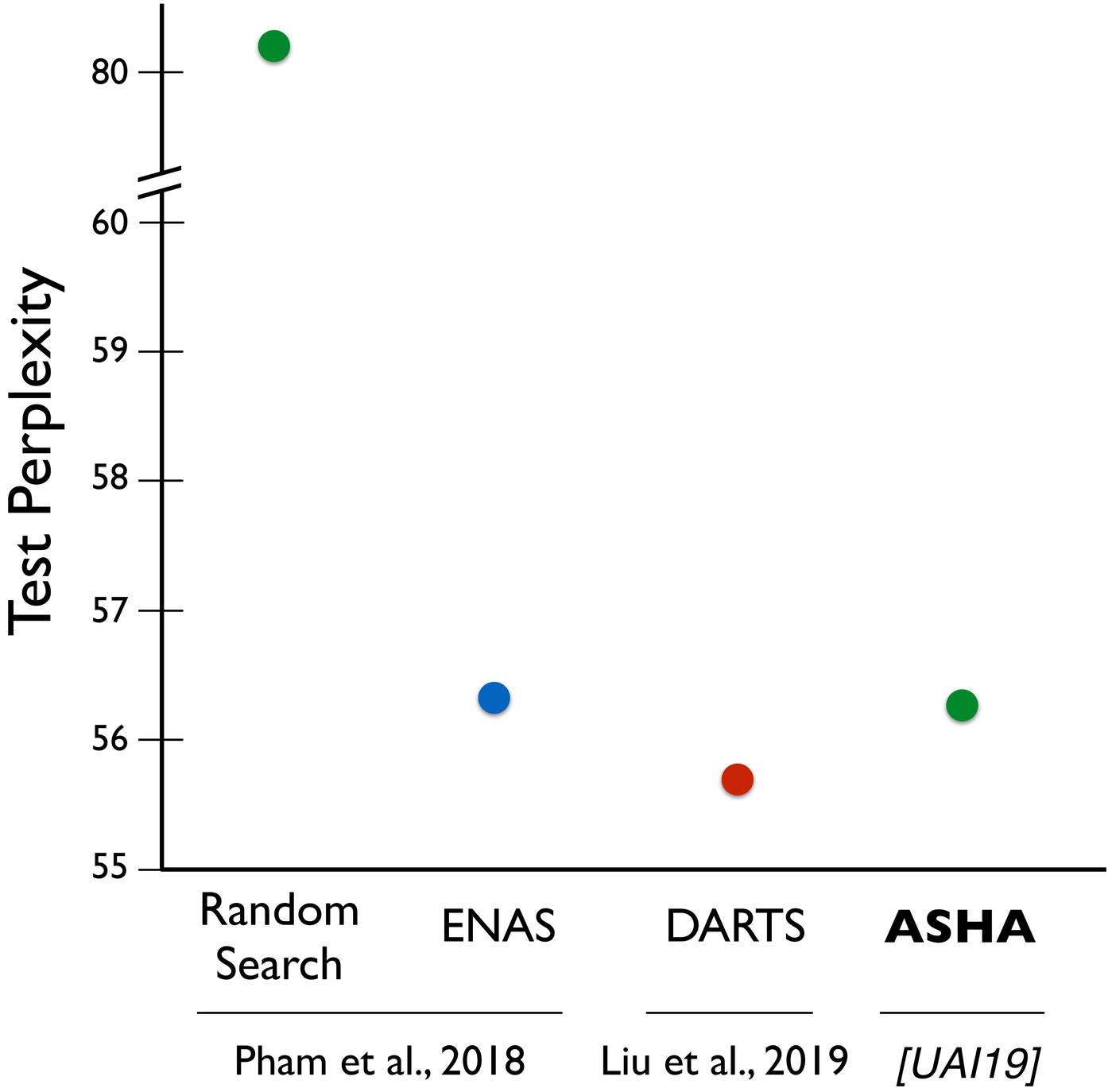
Hyperband / ASHA

[ICLR17, JMLR18, MLSys20]



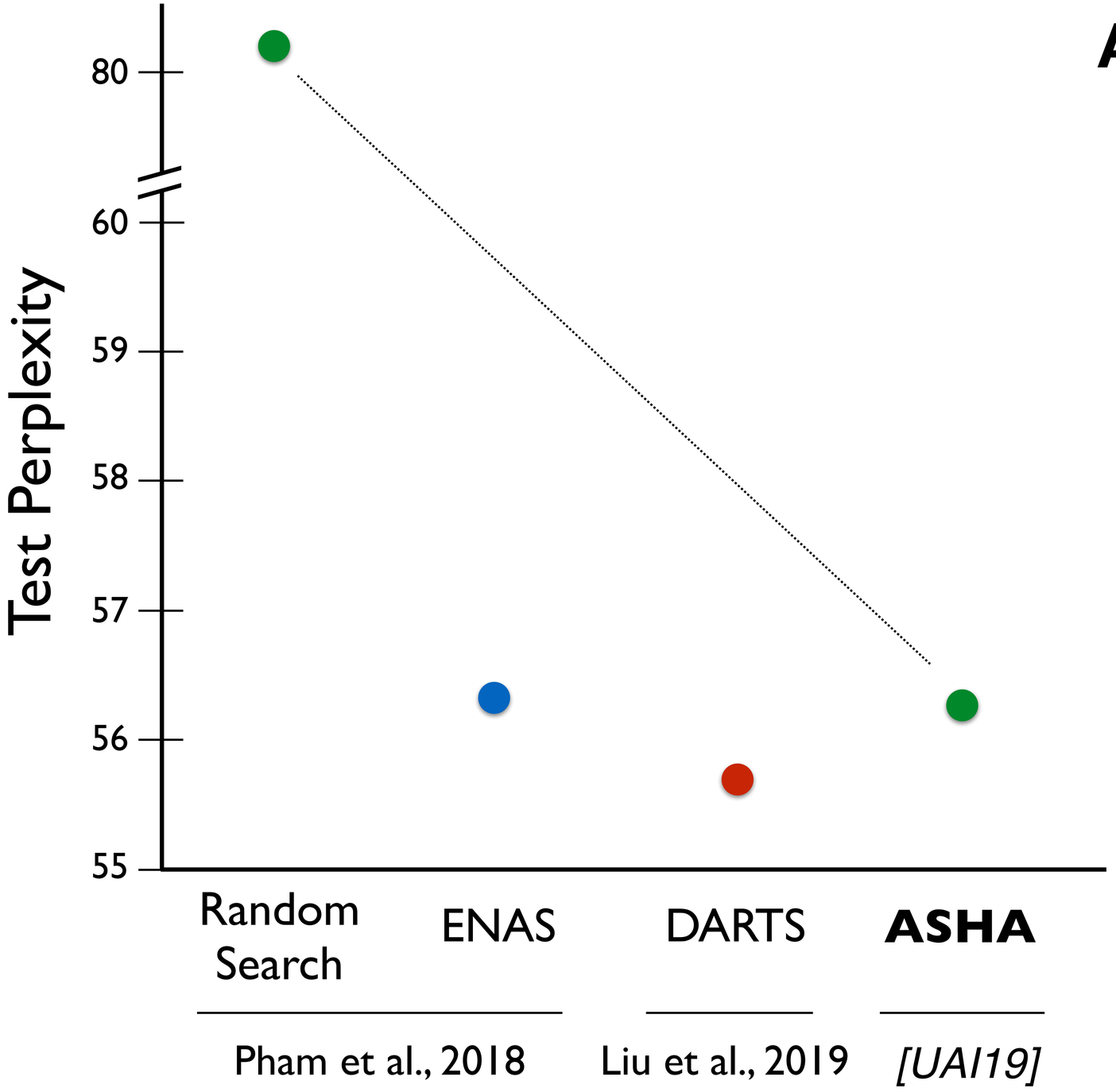
Hyperband / ASHA

[ICLR17, JMLR18, MLSys20]



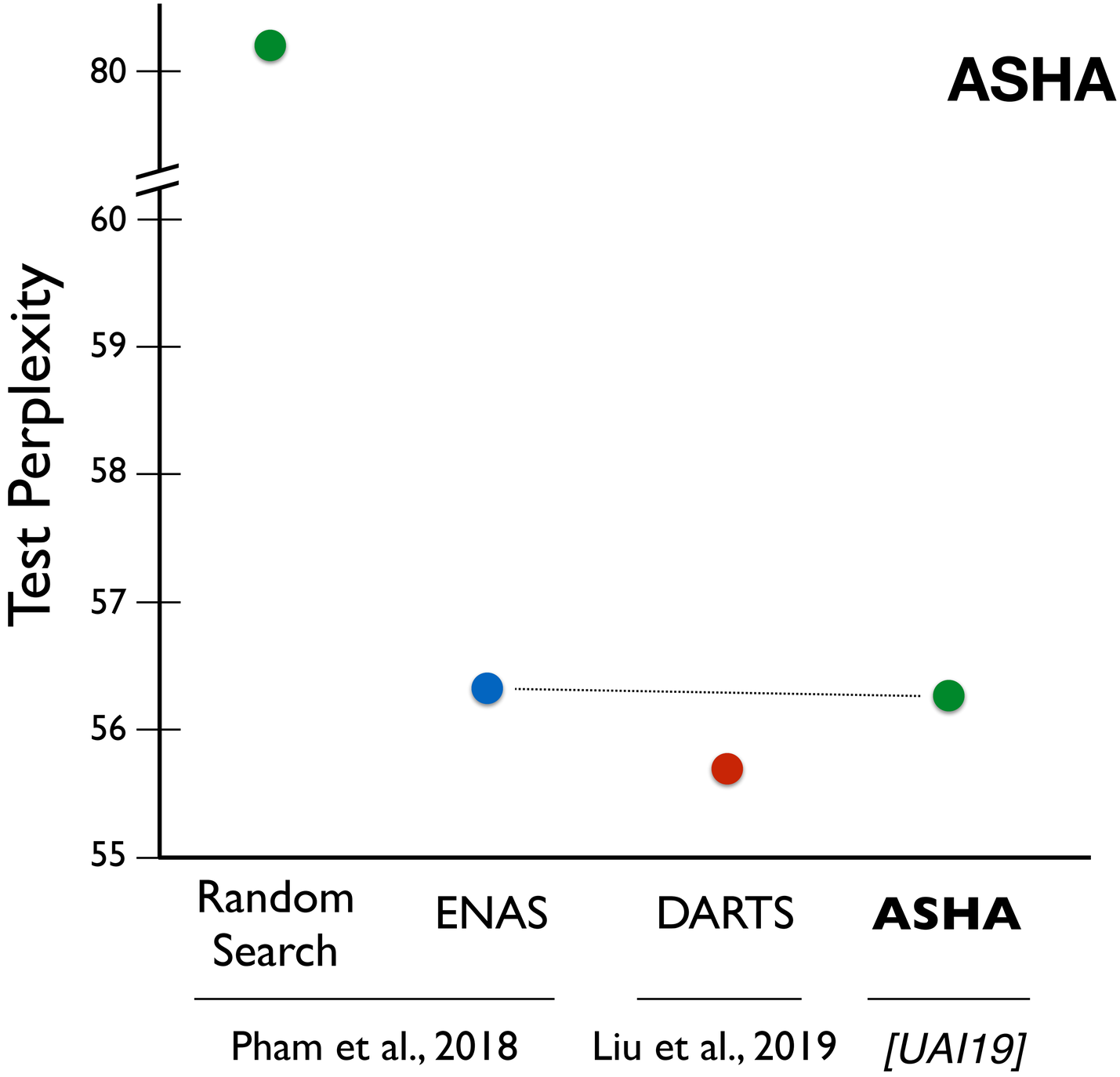
Penn Treebank Benchmark

ASHA dominates previous baselines!



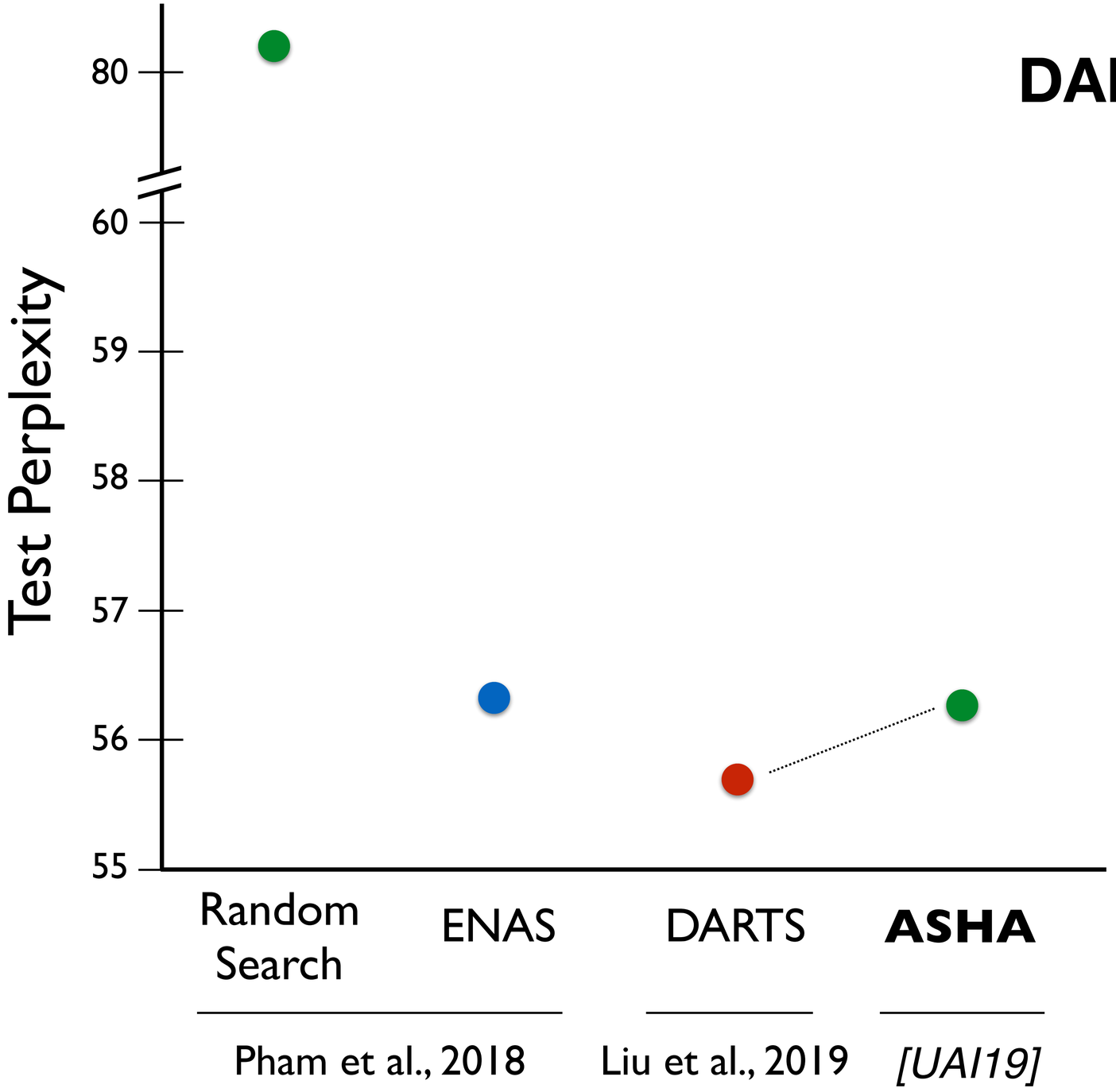
Penn Treebank Benchmark

ASHA is comparable to ENAS!



Penn Treebank Benchmark

DARTS is still better! Why?



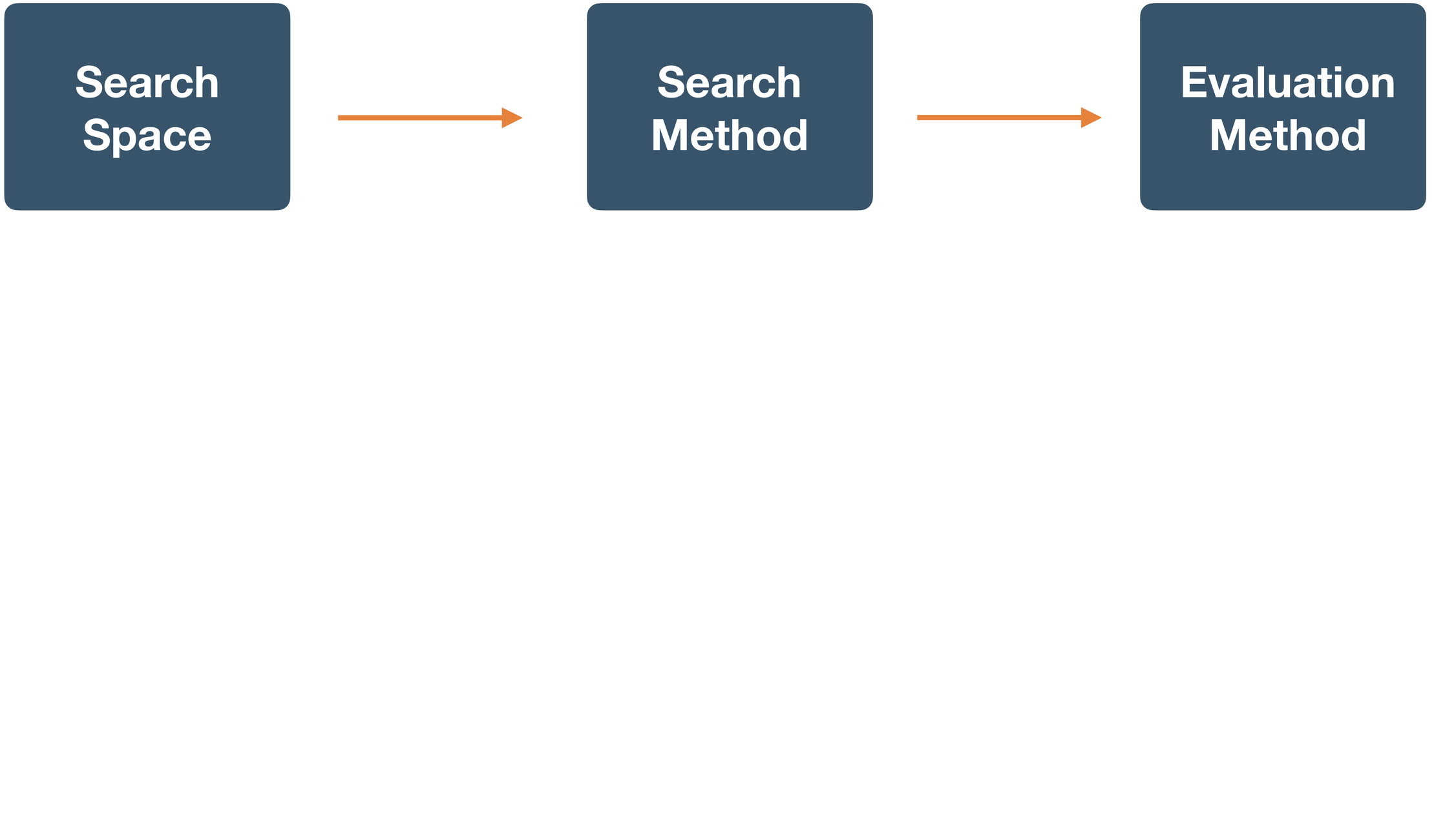
Penn Treebank Benchmark

Q1: How good are NAS heuristics?

Q2: What algorithmic components matter?

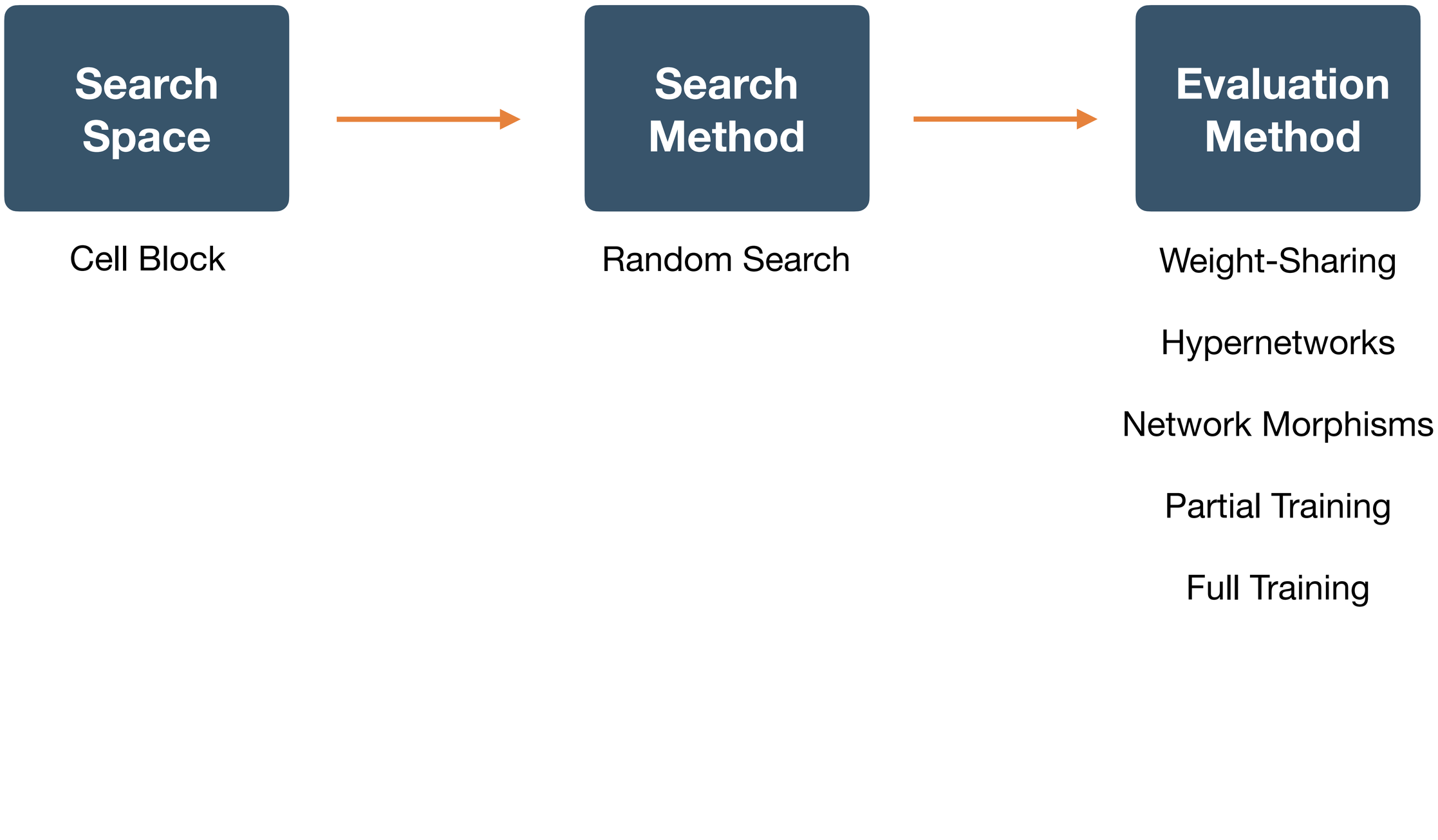
Q3: What about problems beyond NLP / CV?

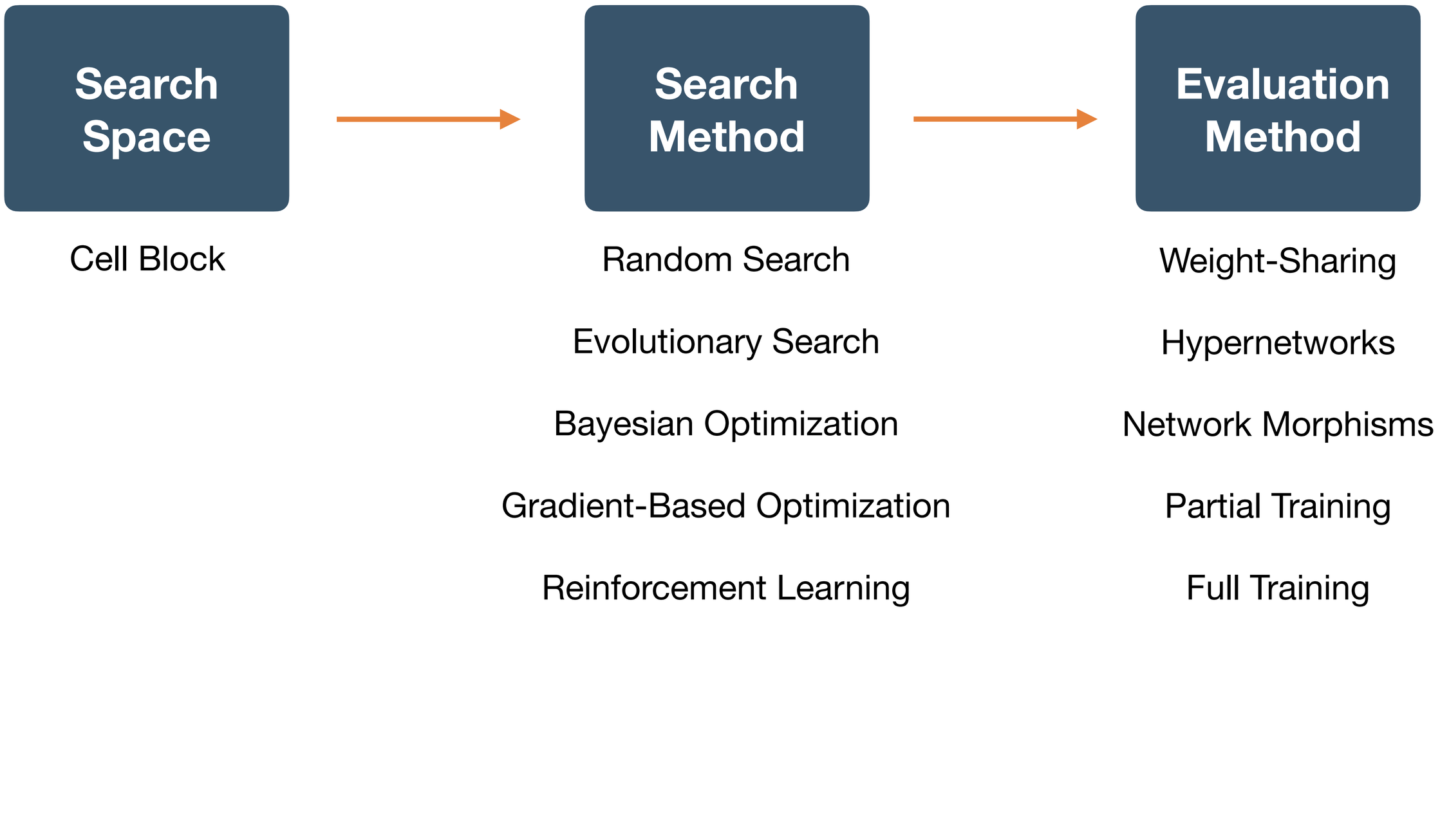
UAI19: L. Li (CMU), A. Talwalkar (CMU, Determined AI)

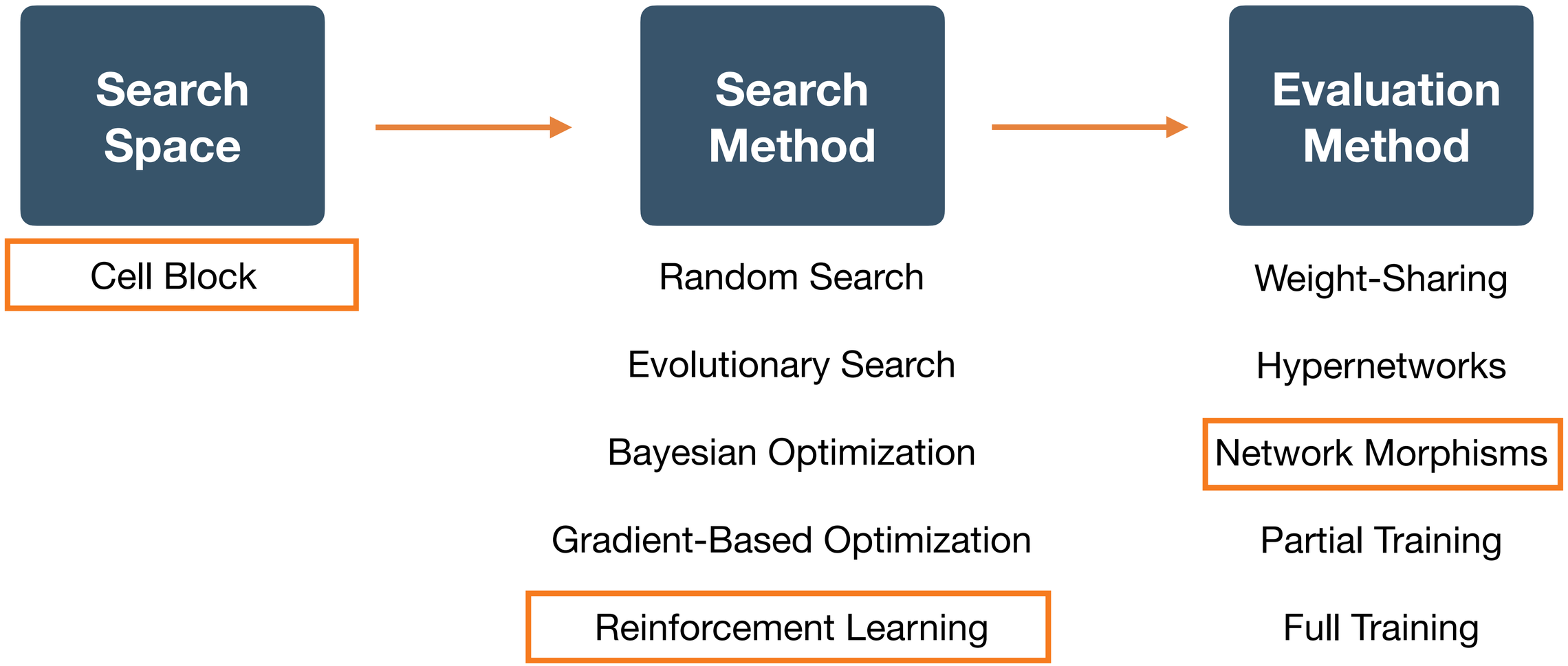

Search Space
Search Method
Evaluation Method
Cheap
Costly

Hyperband / ASHA

[ICLR17, JMLR18, MLSys20]

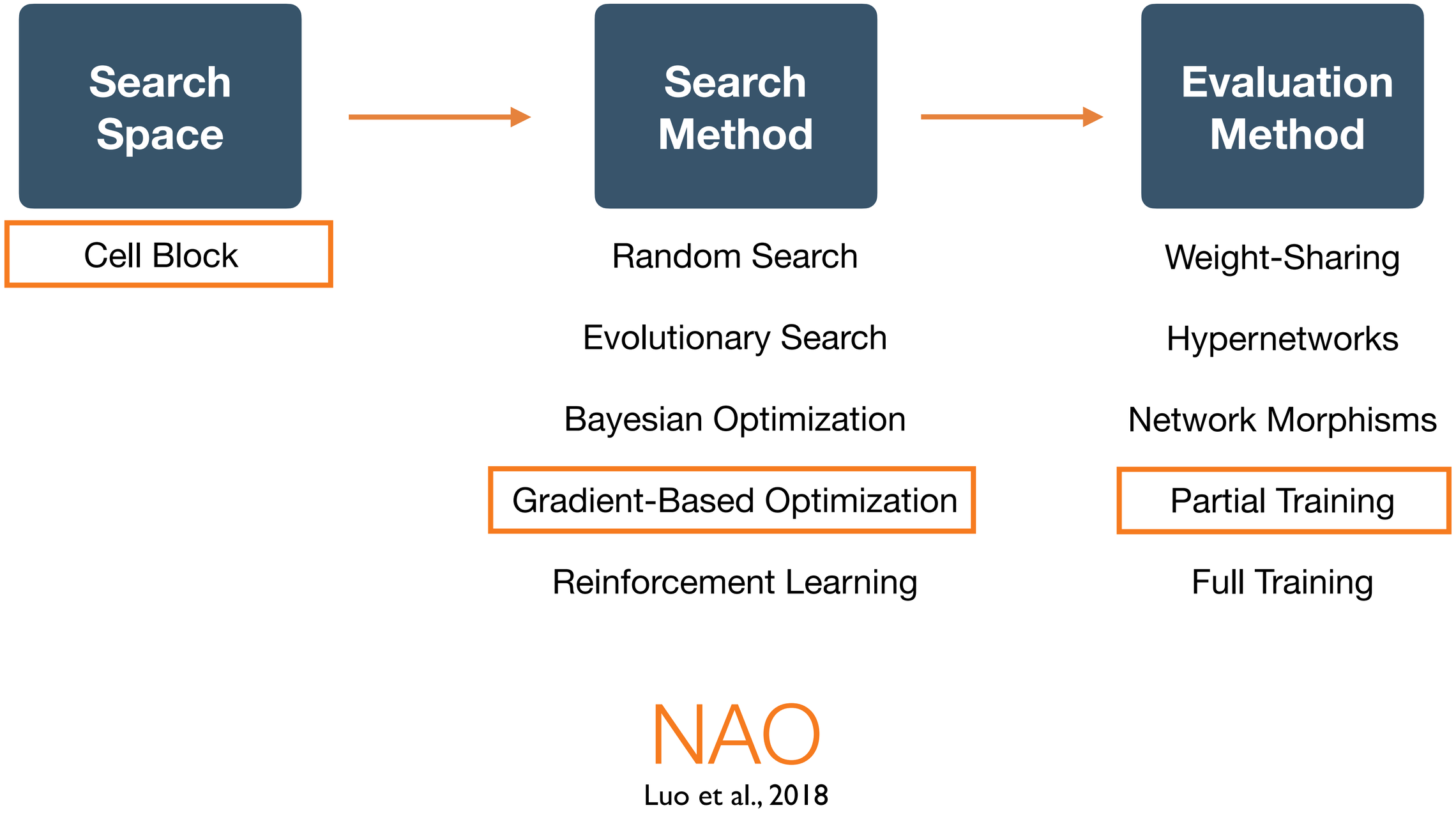

Search Space
Search Method
Evaluation Method
Cell Block
Random Search
Weight-Sharing
Hypernetworks
Network Morphisms
Partial Training
Full Training
Cheap
Costly


Search Space
Search Method
Evaluation Method
Cell Block
Random Search
Evolutionary Search
Bayesian Optimization
Gradient-Based Optimization
Reinforcement Learning
Weight-Sharing
Hypernetworks
Network Morphisms
Partial Training
Full Training
Cheap
Costly



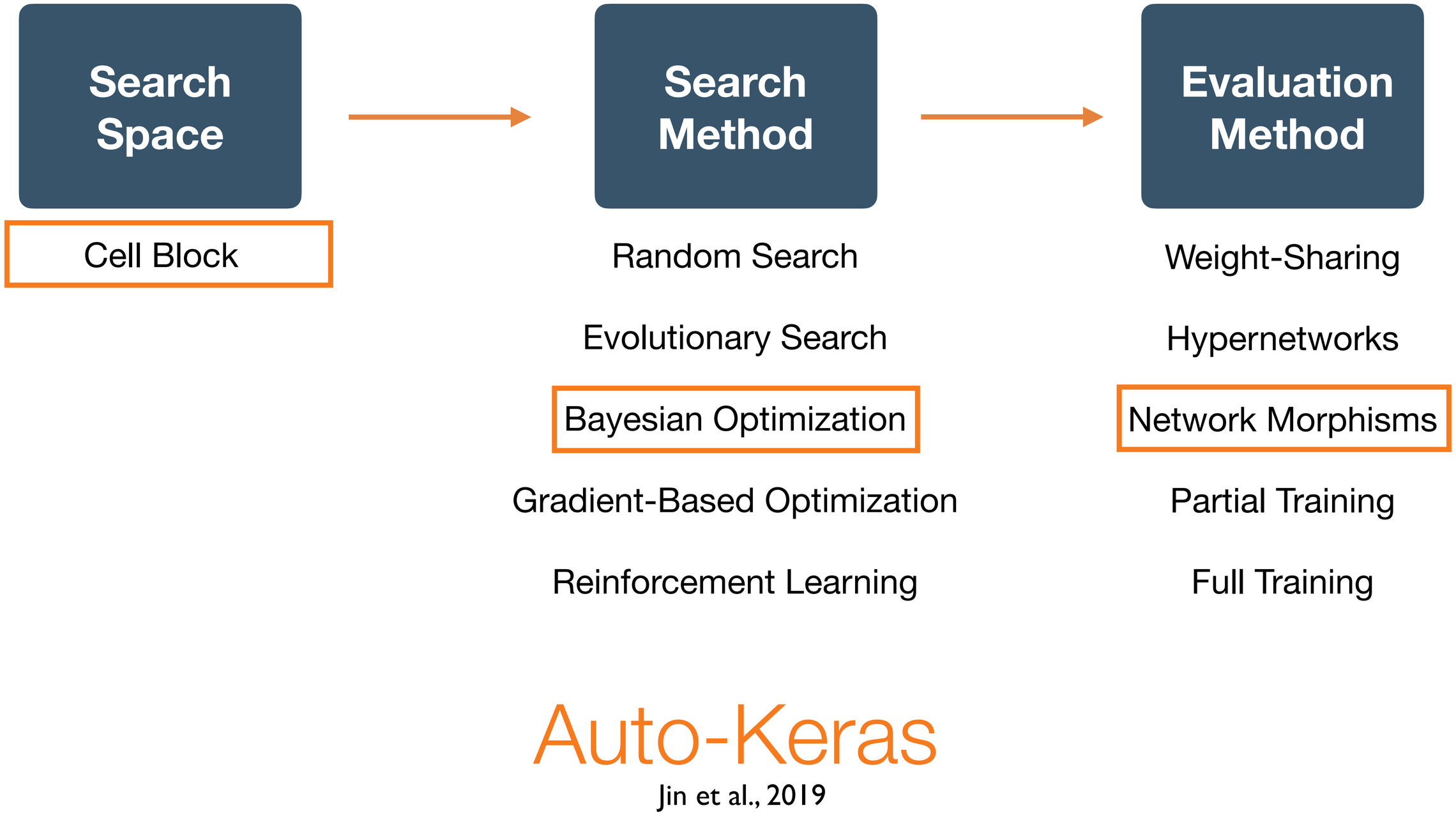
Path-Level

Cai et al., 2018



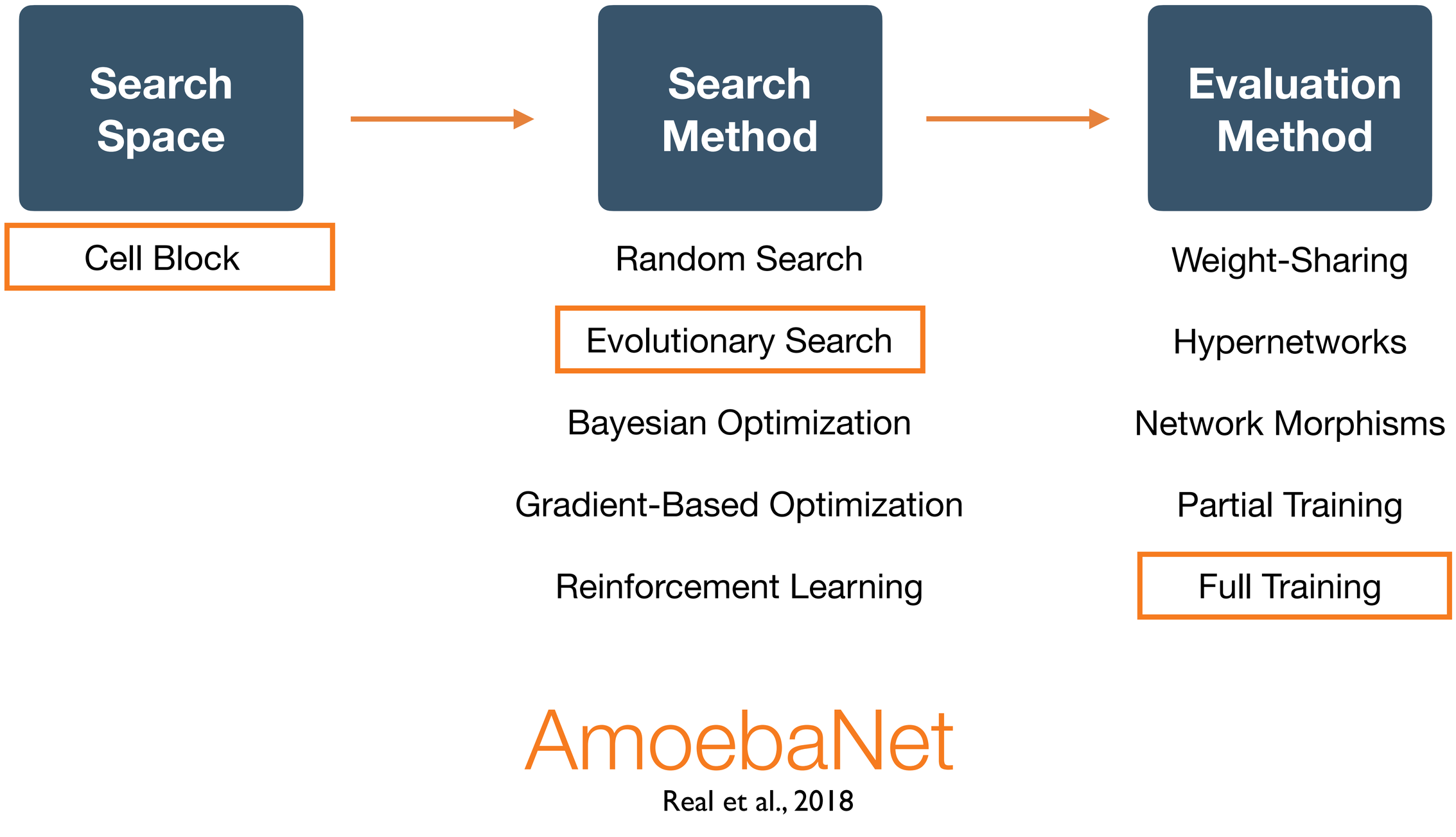
NAO

Luo et al., 2018



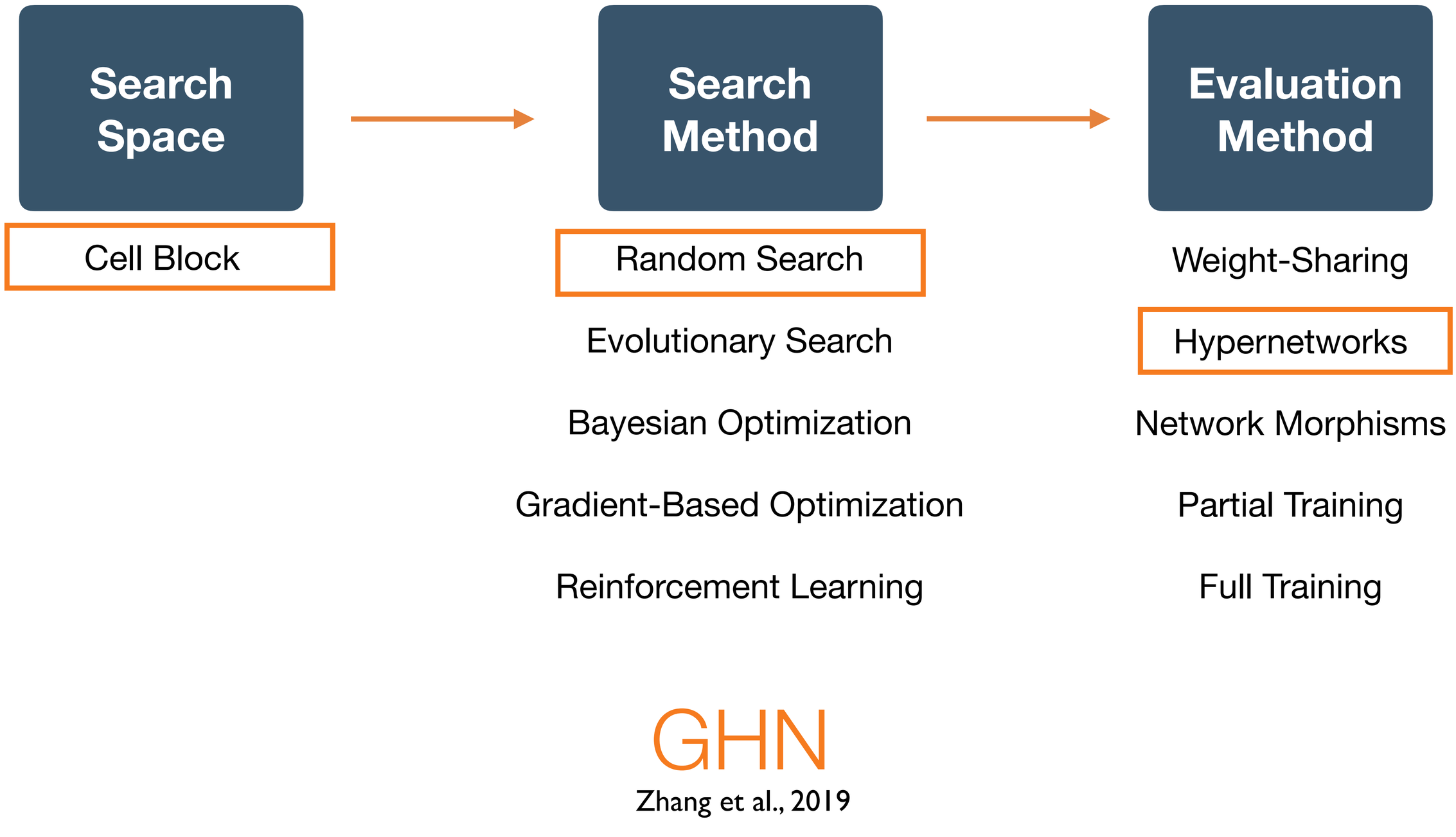
Auto-Keras

Jin et al., 2019



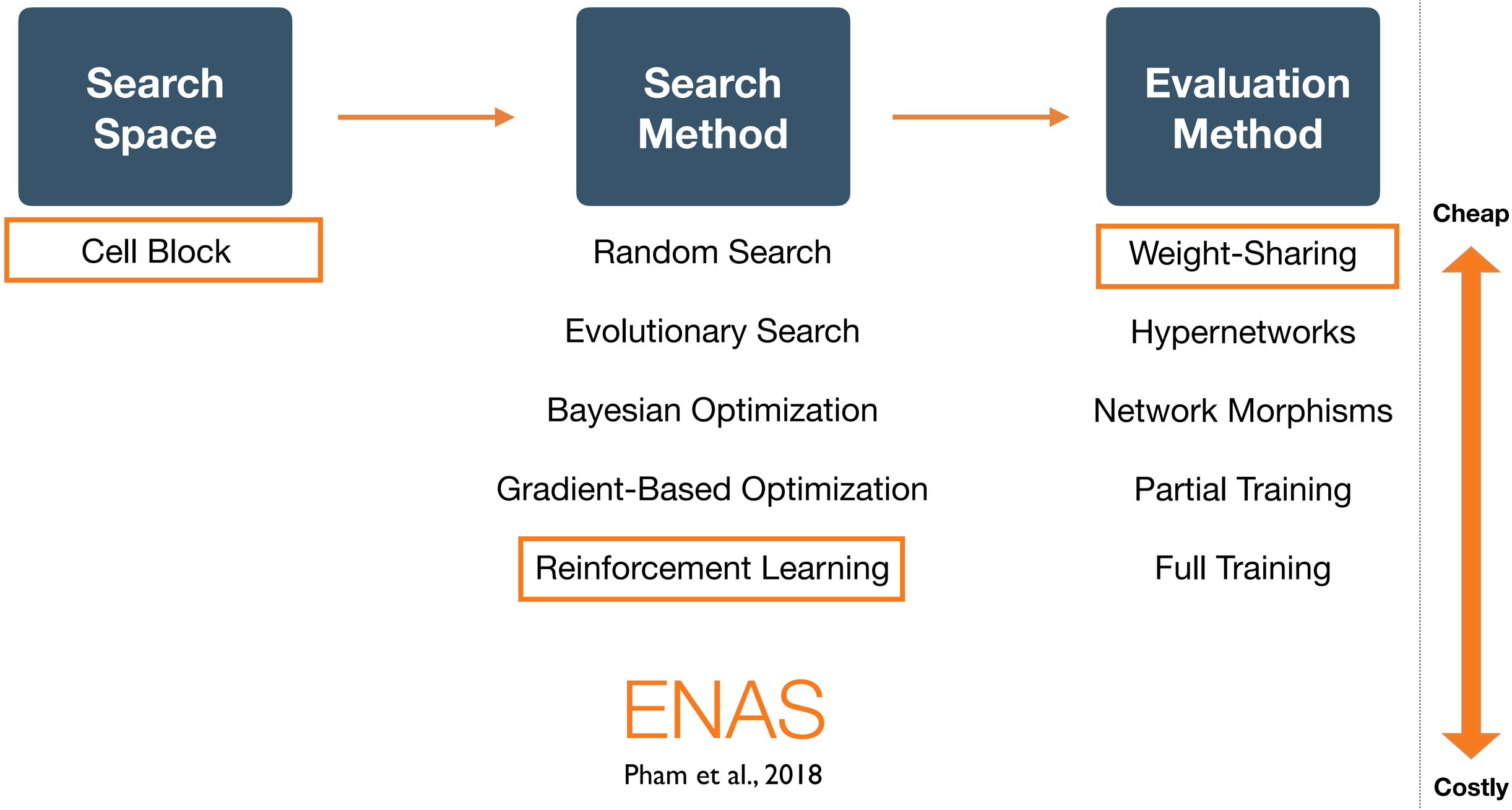
AmoebaNet

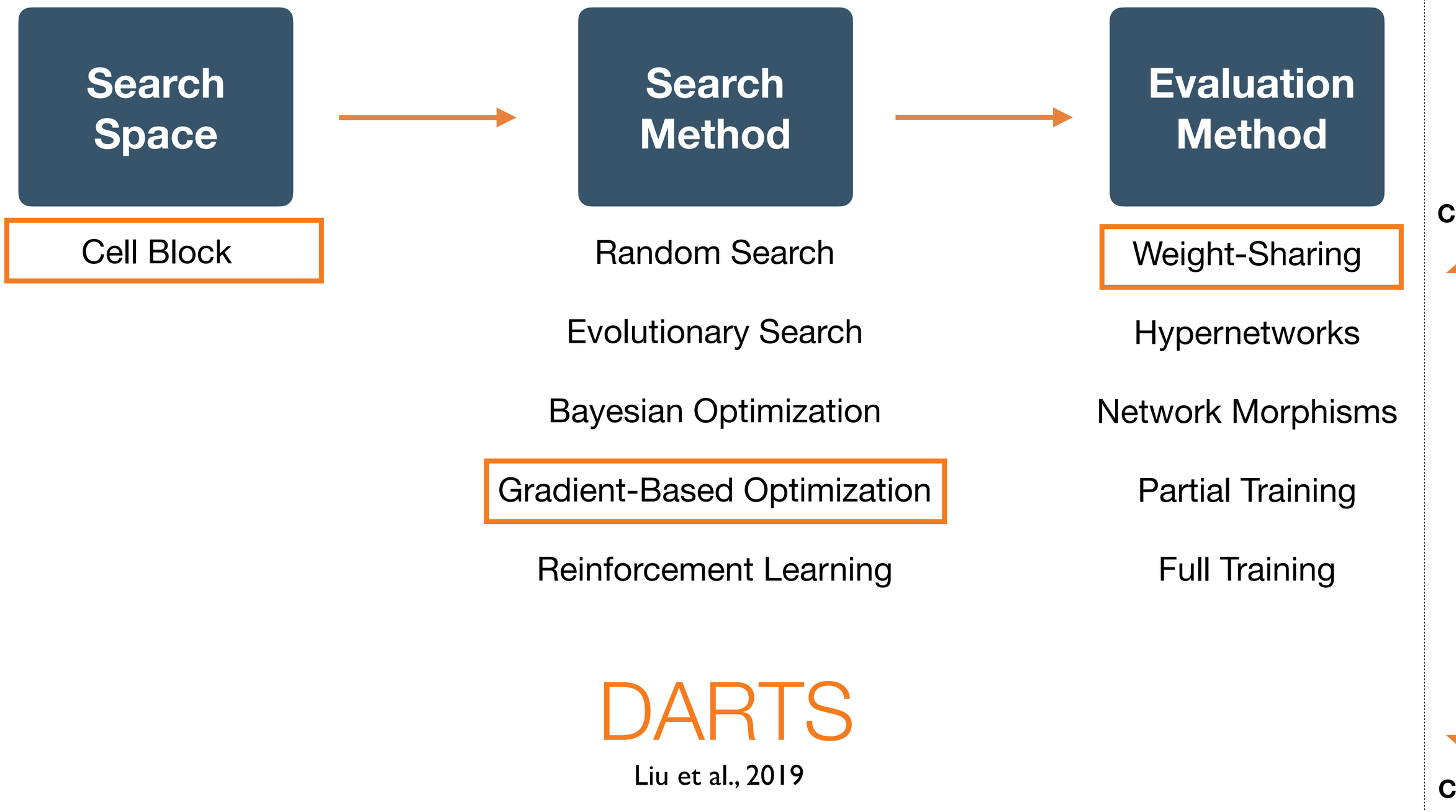
Real et al., 2018



GHN

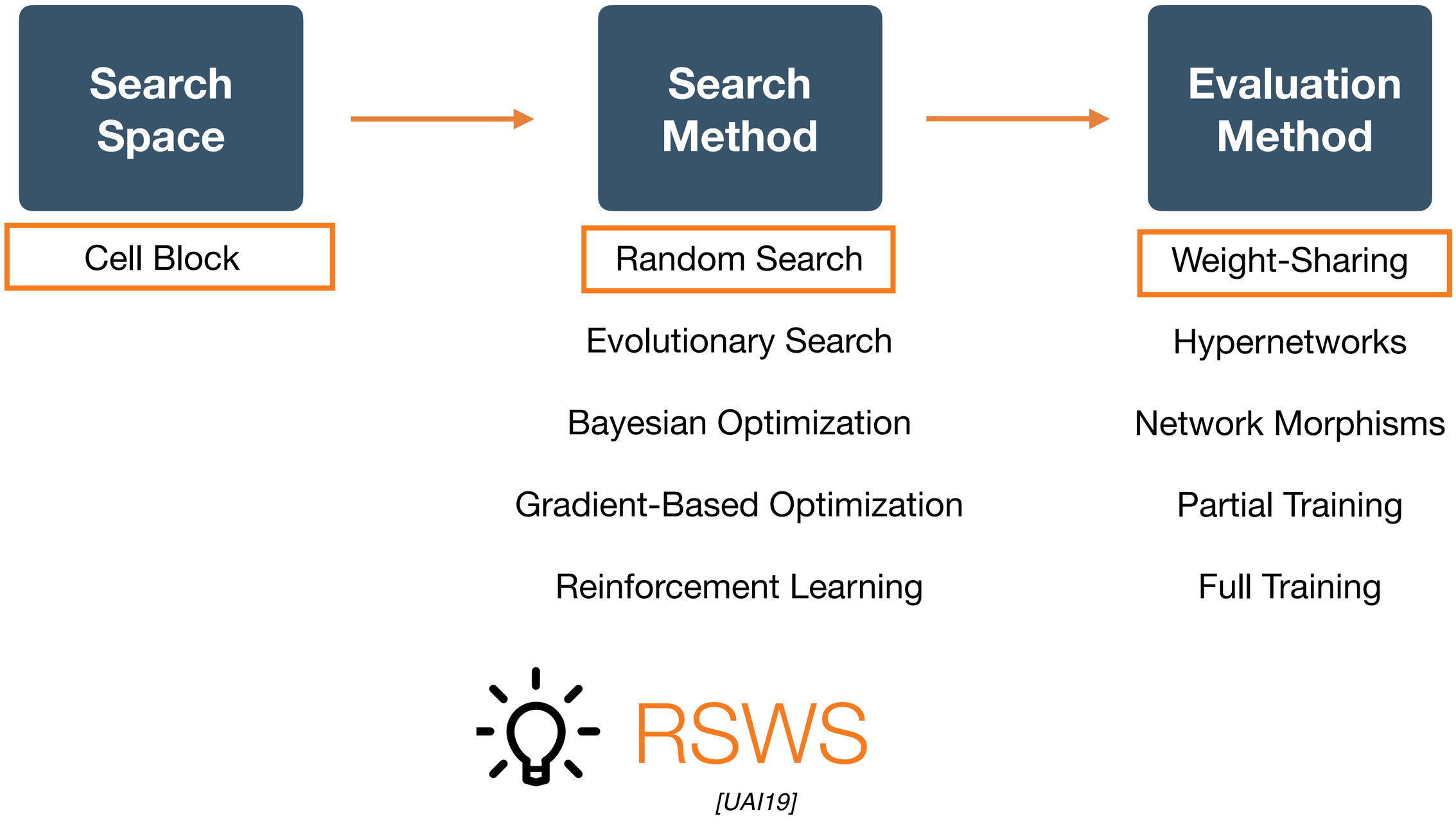
Zhang et al., 2019


Search Space
Search Method
Evaluation Method
Cell Block
Random Search
Evolutionary Search
Bayesian Optimization
Gradient-Based Optimization
Reinforcement Learning
Weight-Sharing
Hypernetworks
Network Morphisms
Partial Training
Full Training
Cheap
Costly
ENAS
Pham et al., 2018



DARTS

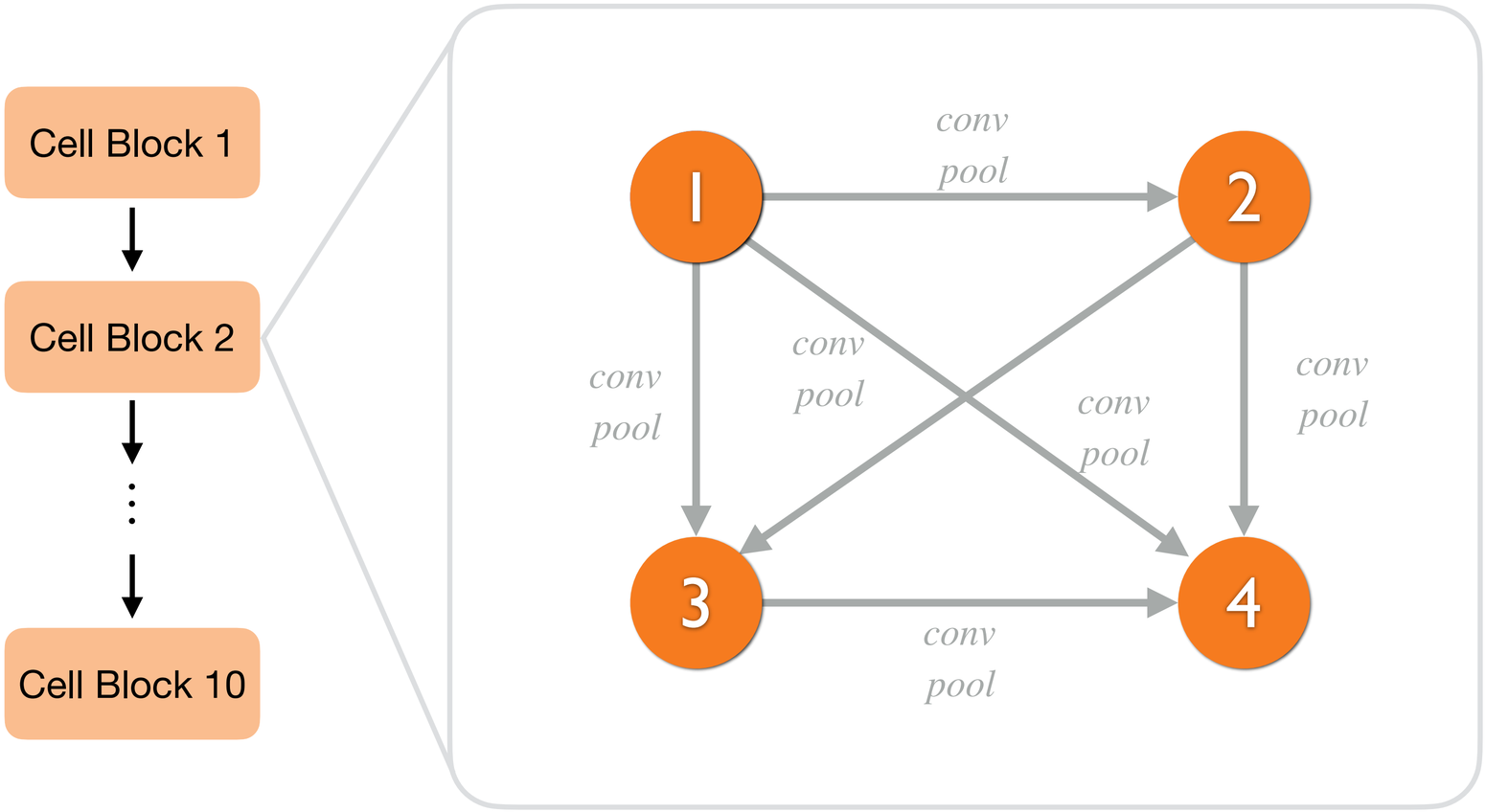
Liu et al., 2019


Search Space
Search Method
Evaluation Method
Cell Block
Random Search
Weight-Sharing
Evolutionary Search
Bayesian Optimization
Gradient-Based Optimization
Reinforcement Learning
Hypernetworks
Network Morphisms
Partial Training
Full Training
Cheap
Costly

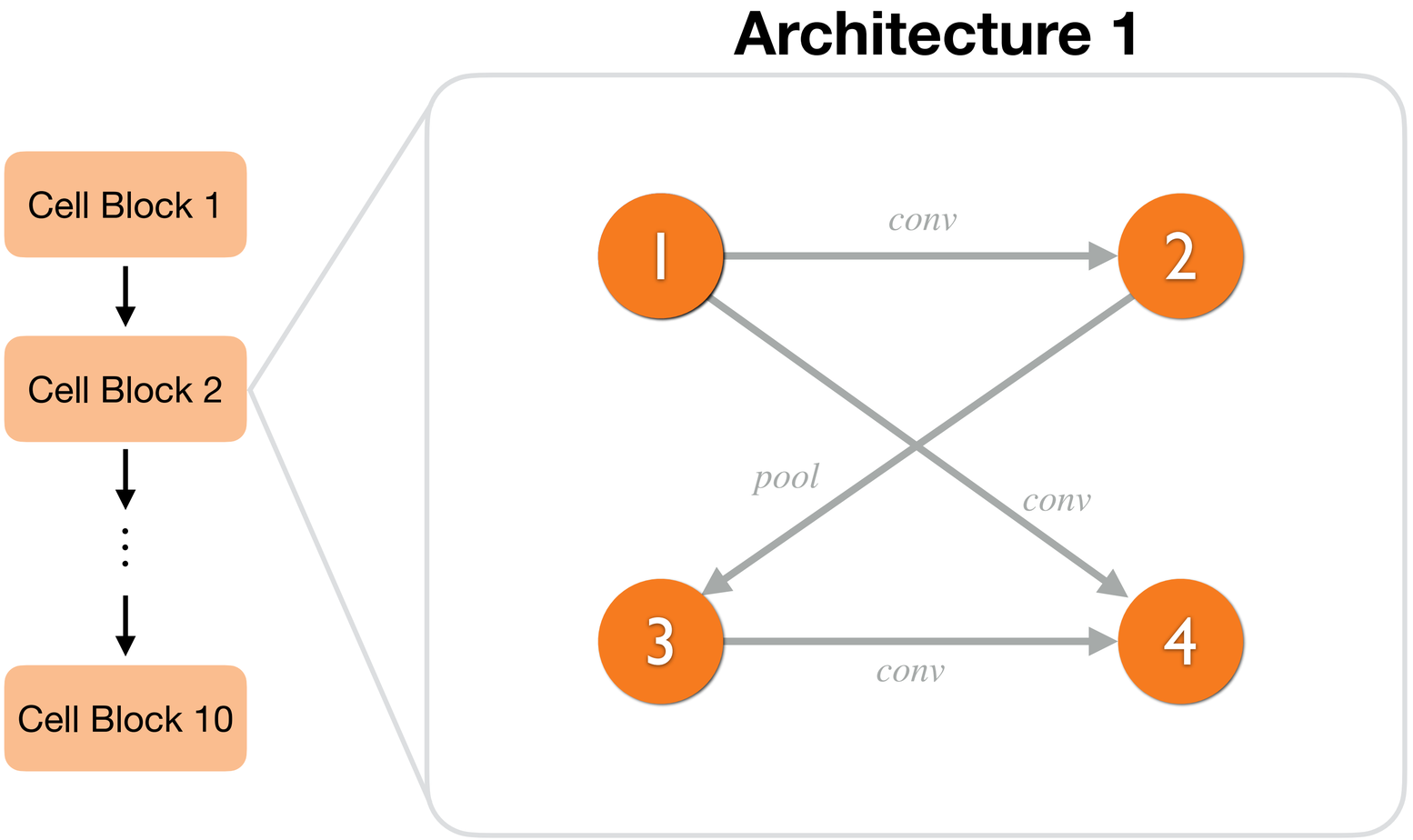
RSWS

[UAI19]

Example: Cell Block & Meta Architecture Space

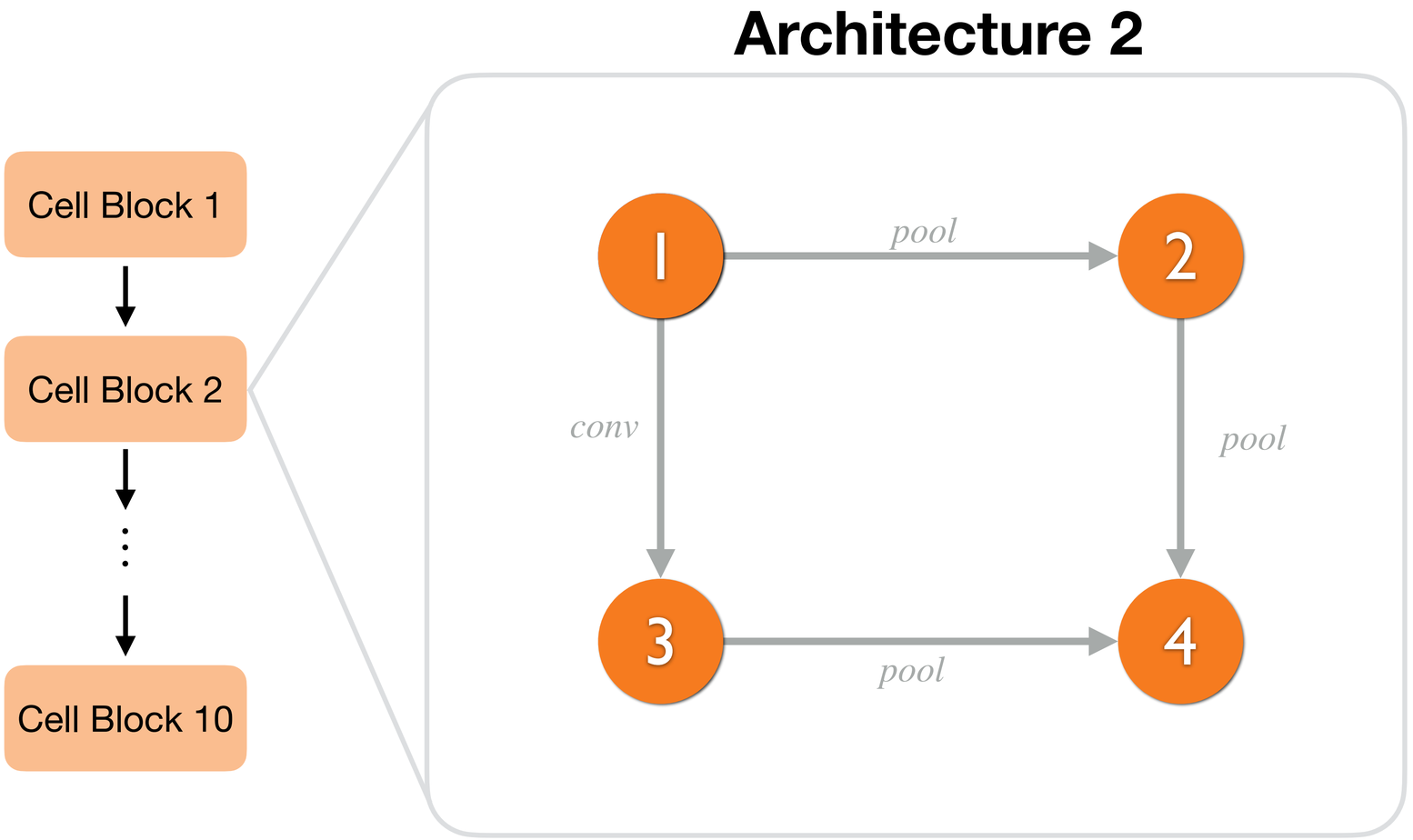


Example: Cell Block & Meta Architecture Space



$$w_1 = \min_{w \in \mathscr{W}} \mathscr{L}\left(\alpha_1(w)\right)$$

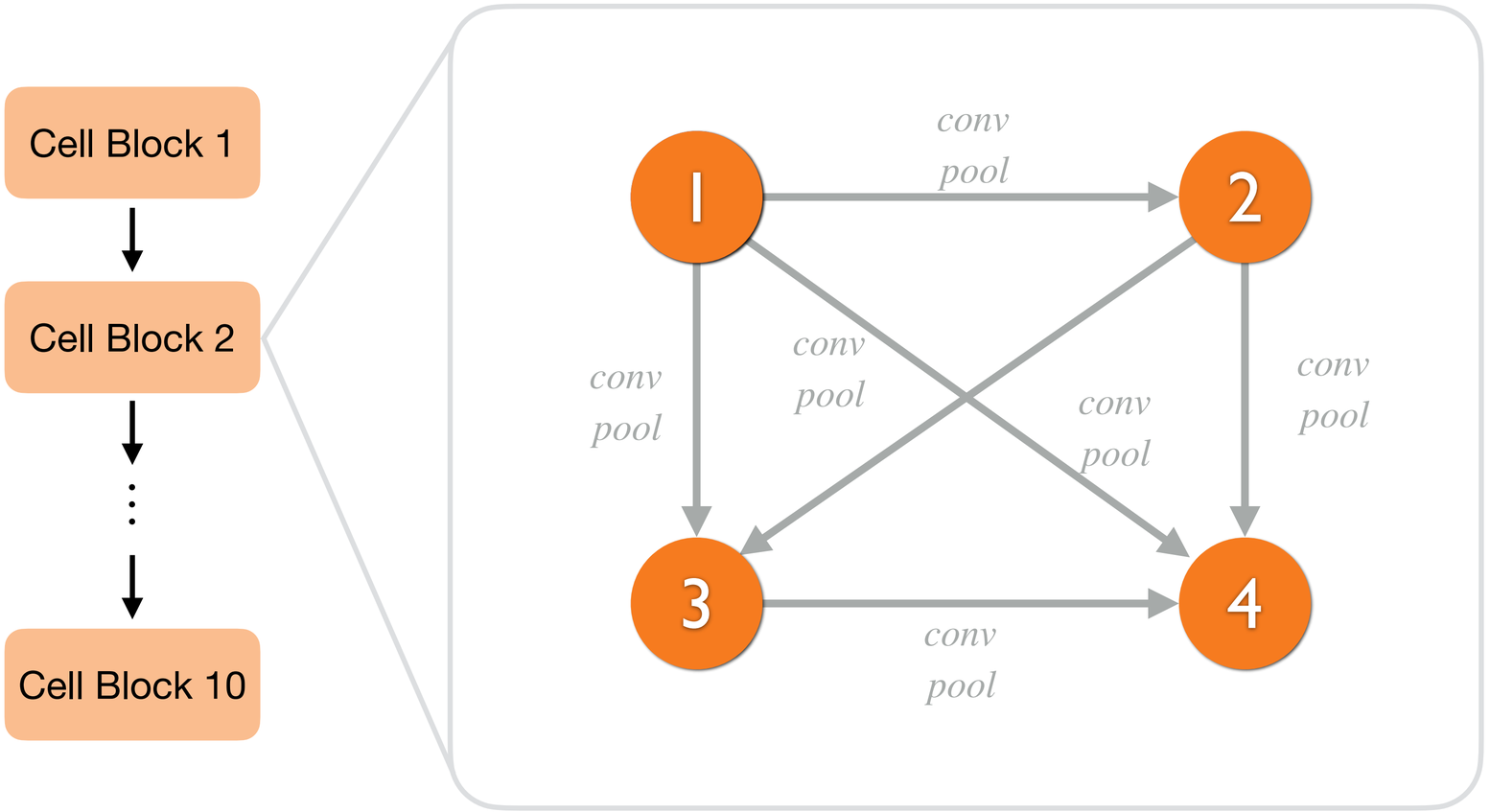
Example: Cell Block & Meta Architecture Space



$$w_1 = \min_{w \in \mathscr{W}} \mathscr{L}\left(\alpha_1(w)\right)$$

$$w_2 = \min_{w \in \mathscr{W}} \mathscr{L}\left(\alpha_2(w)\right)$$

Example: Cell Block & Meta Architecture Space



$$w_1 = \min_{w \in \mathscr{W}} \mathscr{L}\left(\alpha_1(w)\right)$$

$$w_2 = \min_{w \in \mathscr{W}} \mathscr{L}\left(\alpha_2(w)\right)$$

$$\vdots$$

"Full Training"

Train **distinct weights** for each architecture

Underlying Objective

Weights	Goal	Cost	Objective Function
Distinct	optimize '**best**' architecture	**High**	$\min_{\alpha \in \mathcal{A}} \left(\min_{w \in \mathcal{W}} \mathcal{L}\left(\alpha(w)\right) \right)$

Find "best" $\alpha \in \mathcal{A}$

Learn optimal weights w for fixed network α

Random Search Weight-Sharing (RSWS)

Weights	Goal	Cost	Objective Function
Distinct	optimize '**best**' architecture	**High**	$\min_{\alpha \in \mathscr{A}} \left(\min_{w \in \mathscr{W}} \mathscr{L}\left(\alpha(w)\right) \right)$
Shared	optimize '**average**' architecture	**Low**	$\min_{w \in \mathscr{W}} \mathbb{E}_{\alpha \sim U[\mathscr{A}]} [\mathscr{L}(\alpha(w))]$

RSWS Algorithm

Step 1 (Learning): Optimize shared weights by sampling architecture at each SGD step

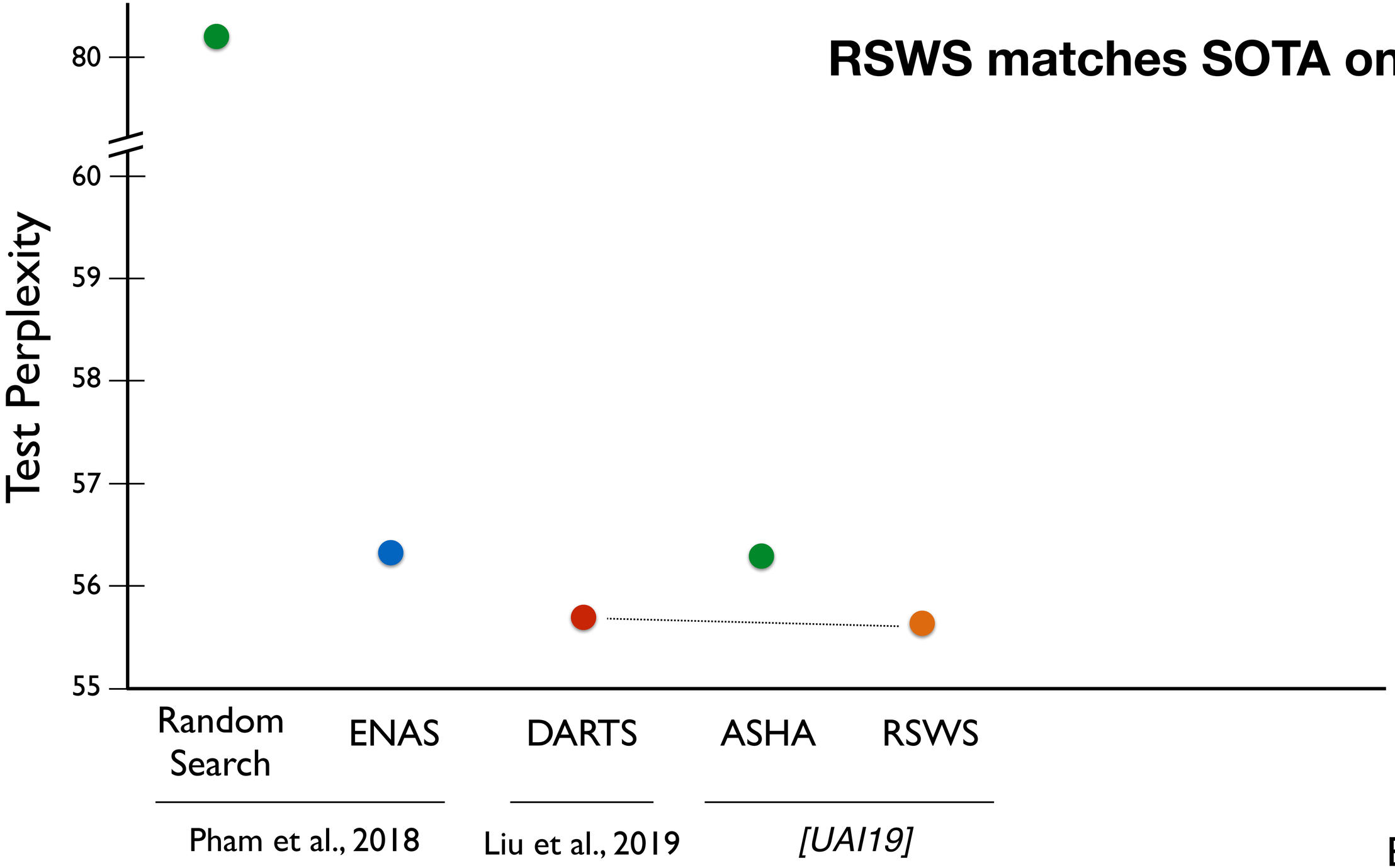
$$\min_{w \in \mathscr{W}} \mathop{\mathbb{E}}_{\alpha \sim U[\mathscr{A}]} [\mathscr{L}(\alpha(w))]$$

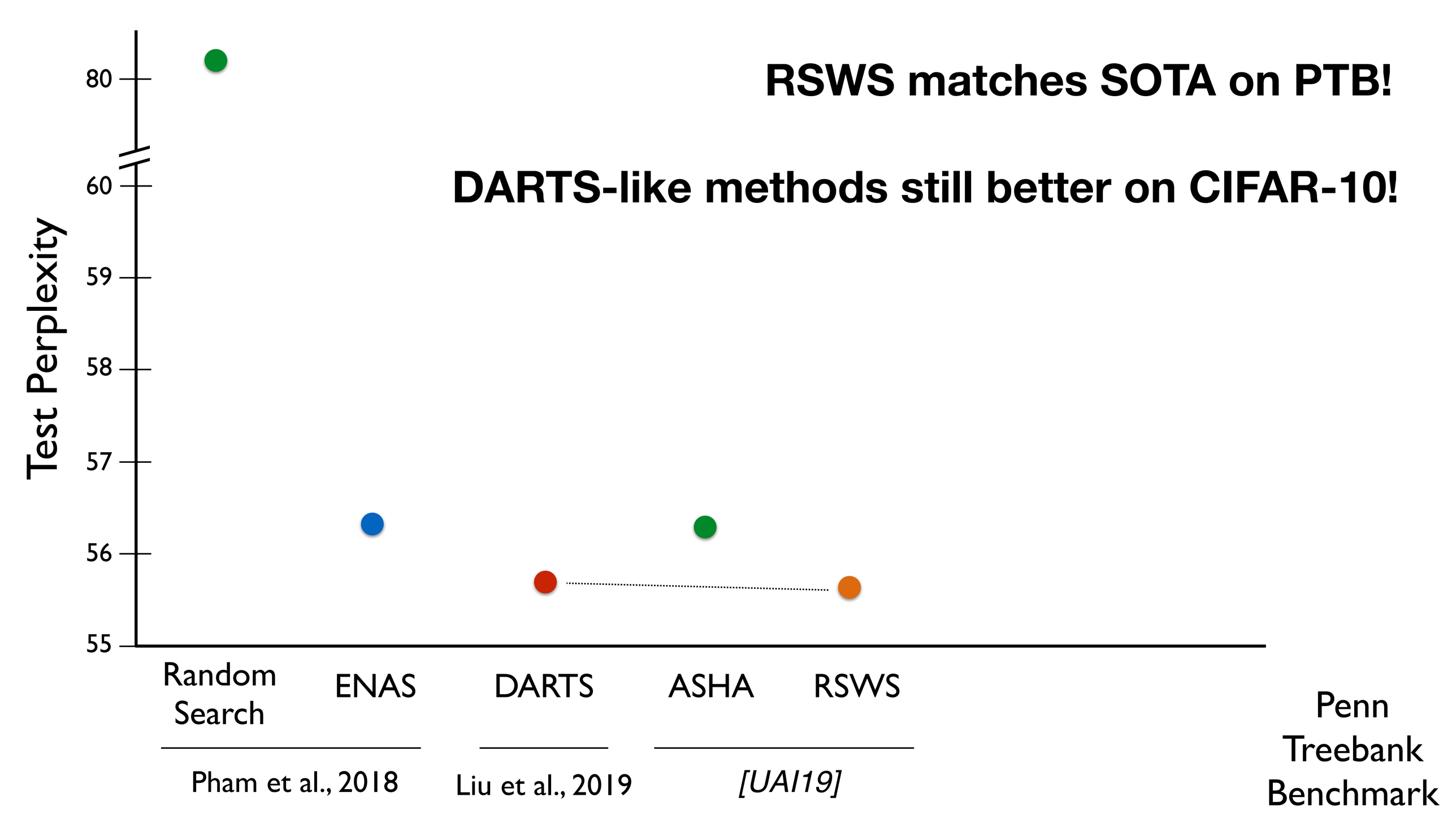
RSWS Algorithm

Step 1 (Learning): Optimize shared weights by sampling architecture at each SGD step

Step 2 (Evaluation): Evaluate random architectures via learned shared weights

RSWS matches SOTA on PTB!



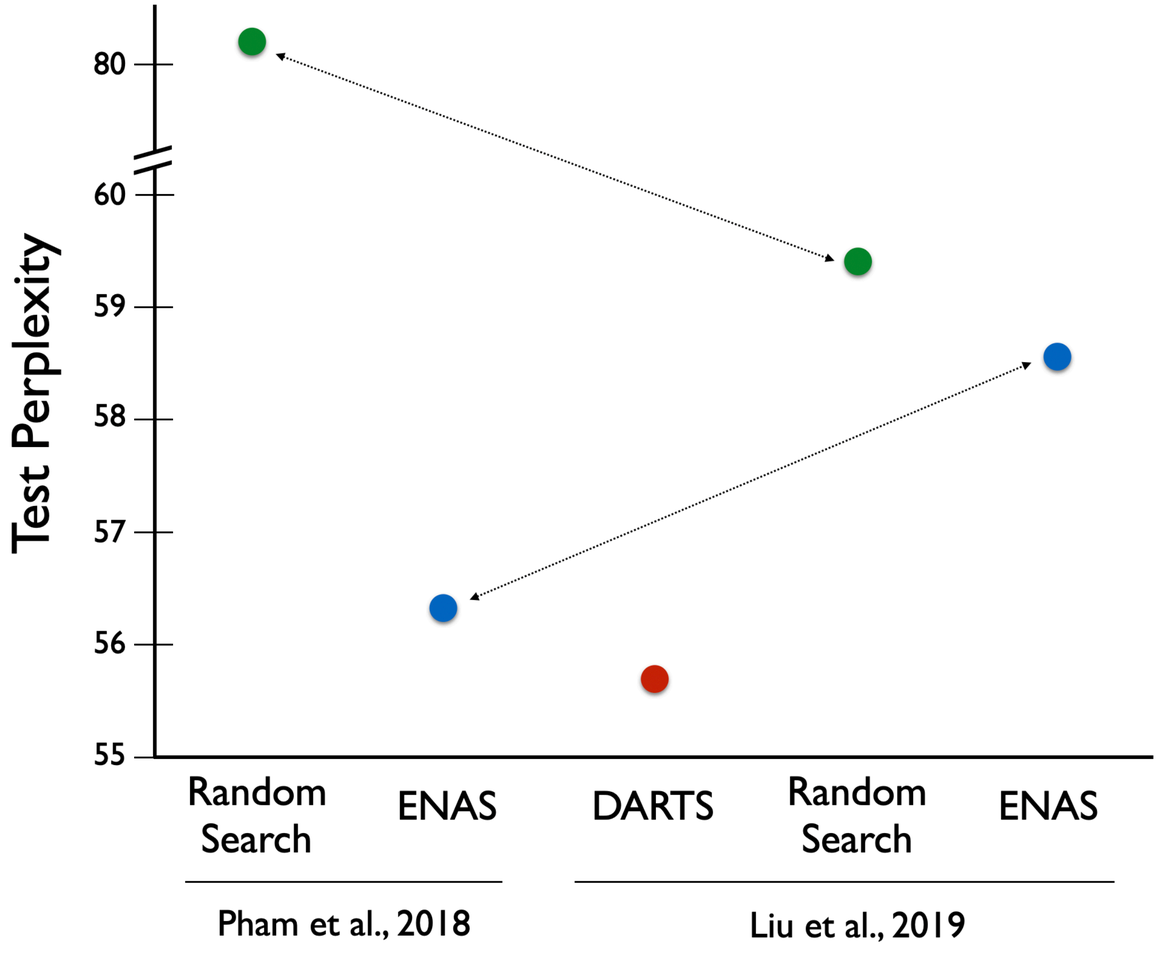

RSWS matches SOTA on PTB!
DARTS-like methods still better on CIFAR-10!
Test Perplexity
80
60
59
58
57
56
55
Random Search
ENAS
DARTS
ASHA
RSWS
Pham et al., 2018
Liu et al., 2019
[UAI19]
Penn Treebank Benchmark

Other aspects of NAS?

Systems + ML: Reproducibility

NAS is facing a reproducibility crisis

[UAI19]



Penn Treebank Benchmark

Systems + ML: Reproducibility

How did I tune non-architectural hyperparameters?

What random seeds did I use?

What versions of PyTorch and CUDA did I use?

What version of the code did I use?

Which checkpoint gave me that result?

Systems + ML: Complex Workflows

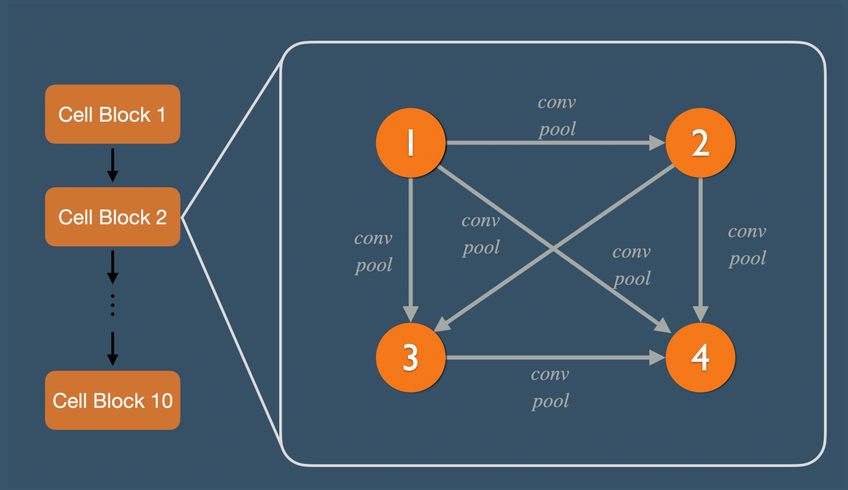
Other Aspects of NAS

Systems + ML

- Lack of Robustness
- Lack of reproducibility
- Complex workflows

Multi-objective NAS

- Optimize for both accuracy and inference constraints



Automation / Efficiency Concerns

- Search spaces rely on expert / domain knowledge
- Computational issues associated with more general search spaces / methods

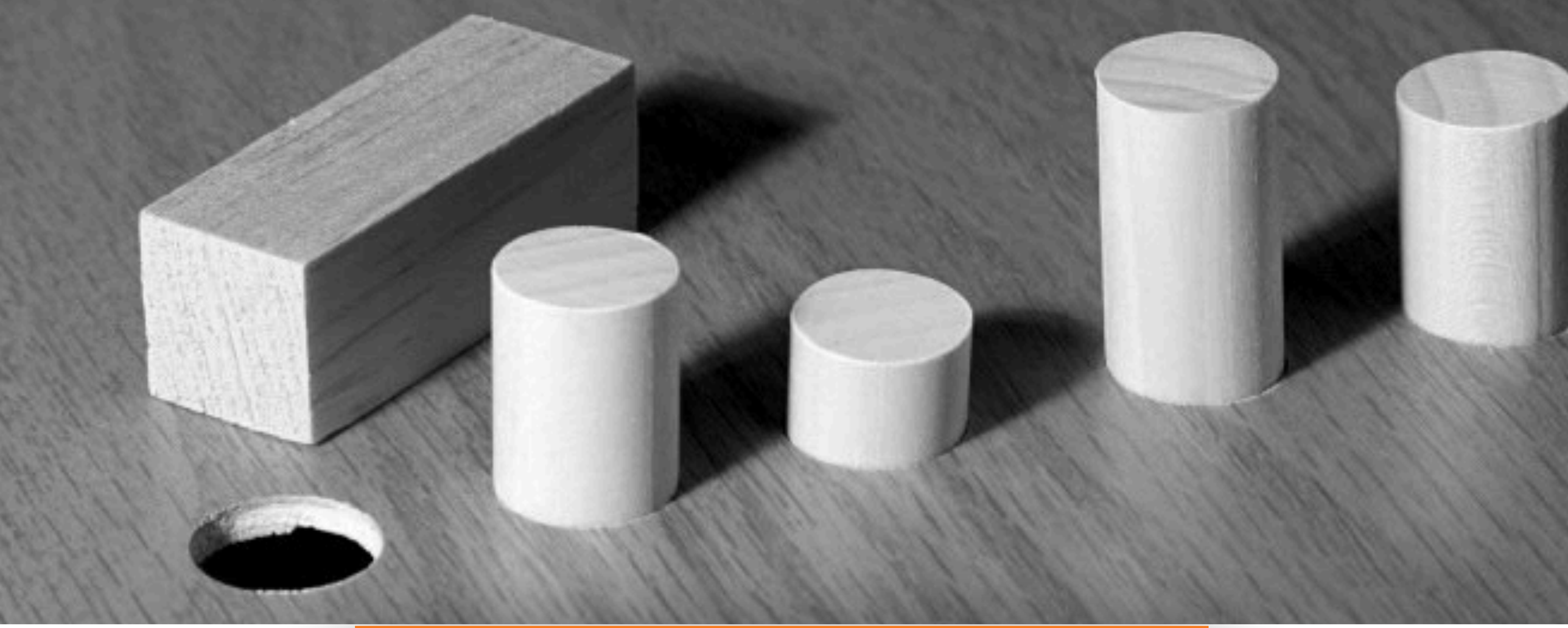
NAS-Bench-360: Benchmarking Diverse Tasks for Neural Architecture Search, R. Tu, M. Khodak, N. Roberts, A. Talwalkar

Q1: How good are NAS heuristics?

Q2: What algorithmic components matter?

Q3: What about problems beyond NLP / CV?

NeurIPS22 Benchmarking Track: R. Tu*, N. Roberts*, M. Khodak, J. Shen, F. Sala, A. Talwalkar



www.nature.com/scientificreports

OPEN

Converting tabular data into images for deep learning with convolutional neural networks

Ingredients for Robust Evaluation

Curate representative suite of diverse tasks

Perform apples-to-apples comparisons

Pick appropriate competitors

NAS-Bench-360: Benchmarking Diverse Tasks

https://nb360.ml.cmu.edu/

Curate representative suite of diverse tasks

Axes of Diversity

- Data dimensionality
- Dataset size
- Problem type
- Application domain

Task name	Size	Dim.	Type	Learning objective	New to NAS
CIFAR-100	60K	2D	Point	Classify natural images into 100 classes	
Spherical	60K	2D	Point	Classify spherically projected images into 100 classes	✓
NinaPro	3956	2D	Point	Classify sEMG signals into 18 classes corresponding to hand gestures	✓
FSD50K	51K	2D	Point (multi-label)	Classify sound events in log-mel spectrograms with 200 labels	✓
Darcy Flow	1100	2D	Dense	Predict the final state of a fluid from its initial conditions	✓
PSICOV	3606	2D	Dense	Predict pairwise distances between residuals from 2D protein sequence features	✓
Cosmic	5250	2D	Dense	Predict propablistic maps to identify cosmic rays in telescope images	✓
ECG	330K	1D	Point	Detecting atrial cardiac disease from a ECG recording via classification	✓
Satellite	1M	1D	Point	Classify satellite image pixels' time series into 24 land cover types	✓
DeepSEA	250K	1D	Point (multi-label)	Predicting chromatin states and binding states of RNA-binding sequences	

Perform apples-to-apples comparisons

Computational settings / baselines

- “Light”: User has resources to train default Wide ResNet (WRN)
- “Heavy”: User can perform hyperparameter tuning on WRN

Evaluation Metrics

- ‘Performance Profiles’ are robust to outlier problems and to small differences among methods

Pick appropriate competitors

Search Space	Search Method	Computational Profile	Specificity
Wide ResNet	n/a	Light	General
DenseNAS-R1	DenseNAS-search	Light	General
Wide ResNet	ASHA	Heavy	General
DARTS	GAEA	Heavy	General

Pick appropriate competitors

Search Space	Search Method	Computational Profile	Specificity
Wide ResNet	n/a	Light	General
DenseNAS-R1	DenseNAS-search	Light	General
Wide ResNet	ASHA	Heavy	General
DARTS	GAEA	Heavy	General
Amber	ENAS	Medium	1D data
AutoDL	DARTS	Medium	Dense Prediction

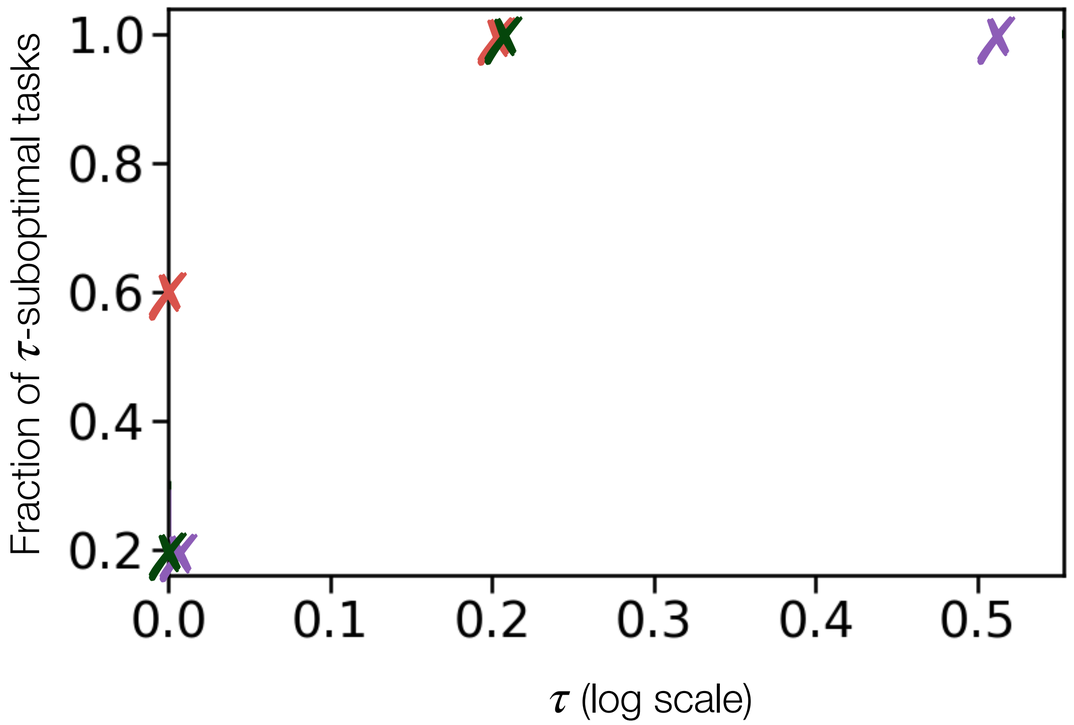
Pick appropriate competitors

Search Space	Search Method	Computational Profile	Specificity
Wide ResNet	n/a	Light	General
DenseNAS-R1	DenseNAS-search	Light	General
Wide ResNet	ASHA	Heavy	General
DARTS	GAEA	Heavy	General
Amber	ENAS	Medium	1D data
AutoDL	DARTS	Medium	Dense Prediction
Hand Designed	n/a	n/a	n/a

Pick appropriate competitors

Search Space	Search Method	Computational Profile	Specificity
Wide ResNet	n/a	Light	General
DenseNAS-R1	DenseNAS-search	Light	General
Wide ResNet	ASHA	Heavy	General
DARTS	GAEA	Heavy	General
Amber	ENAS	Medium	1D data
AutoDL	DARTS	Medium	Dense Prediction
Hand Designed	n/a	n/a	n/a
DASH	**GAEA**	**Light**	**General**

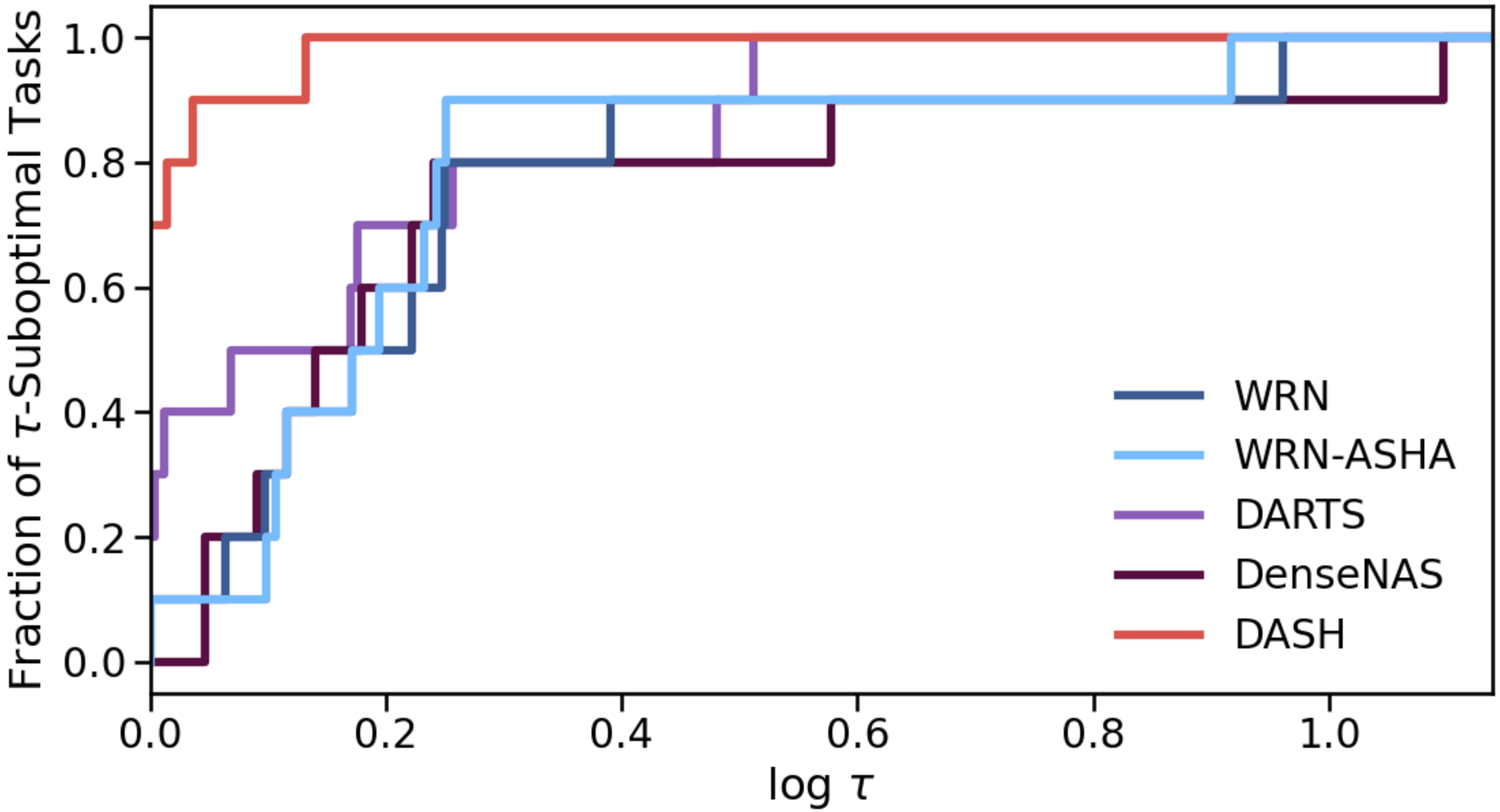
Performance Profiles Summary

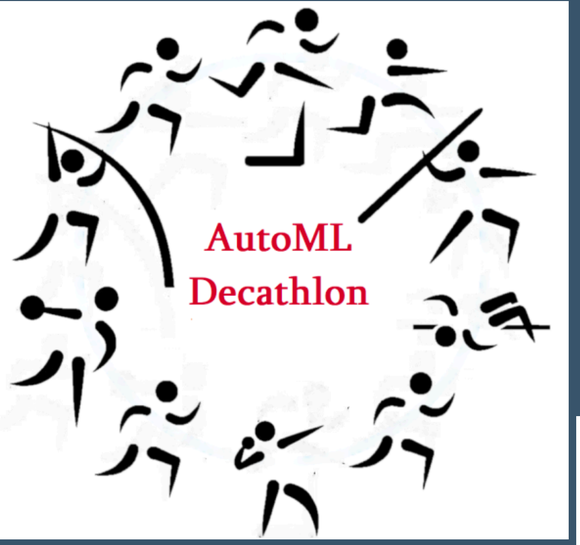


Normalize metrics for different tasks
(0-1 error, MAP, F1 score, MAE_8, …)

Compute fraction of tasks where a method has distance from optimality less than τ

Performance Profiles Summary





Development

July 13 — Oct 19

Participants will refine their methods using public development tasks to estimate performance

Evaluation

Final submissions will be evaluated on a distinct set of private test tasks

Final Results

Nov 11

Final results and analyses will be announced; the winning team will receive a $15,000 prize