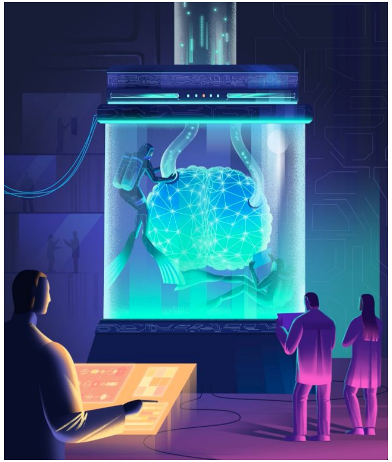
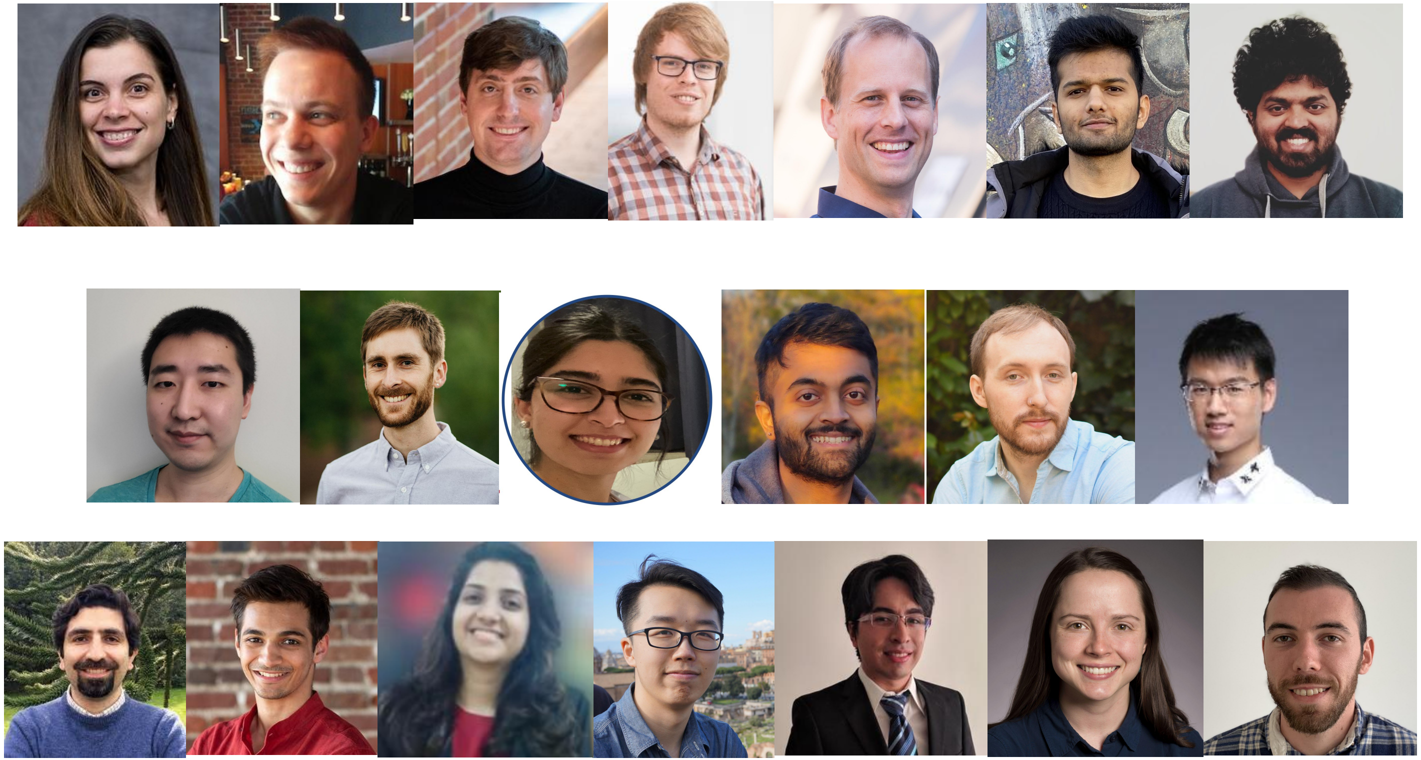
Neural Architecture Search: The Next Frontier



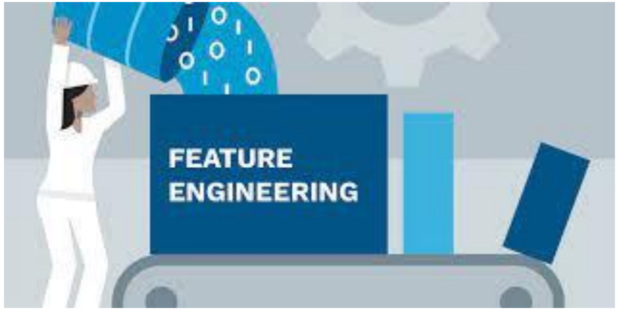
Colin White
colin@abacus.ai
https://crwhite.ml/



Slides (with hyperlinks): https://crwhite.ml/

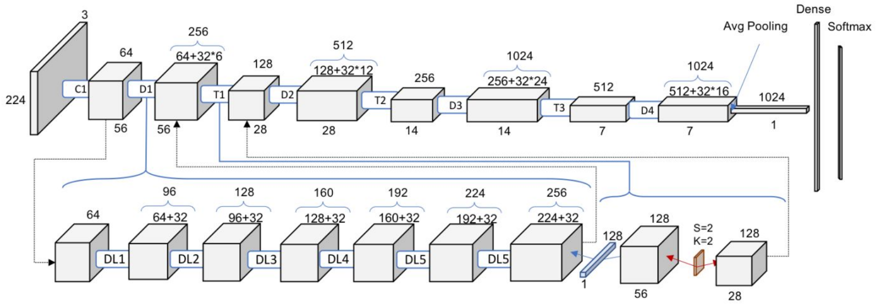
1950s

2012

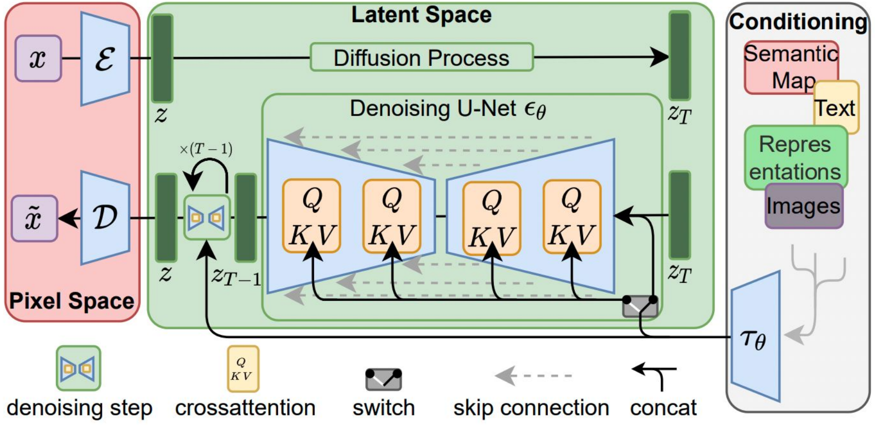


2017

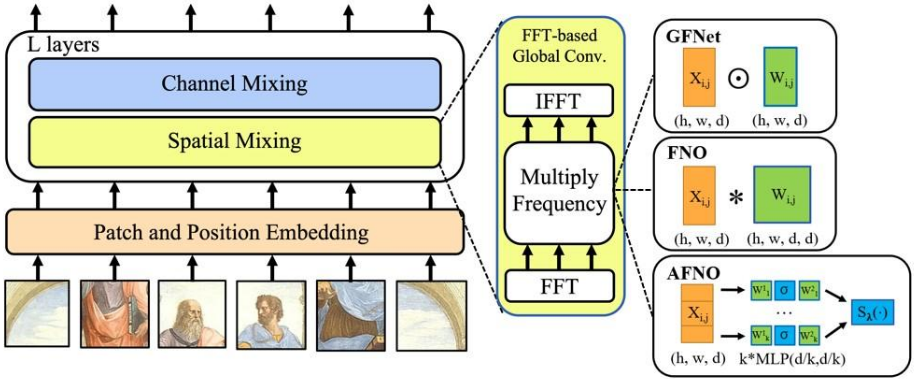
2023



DenseNet (2016)

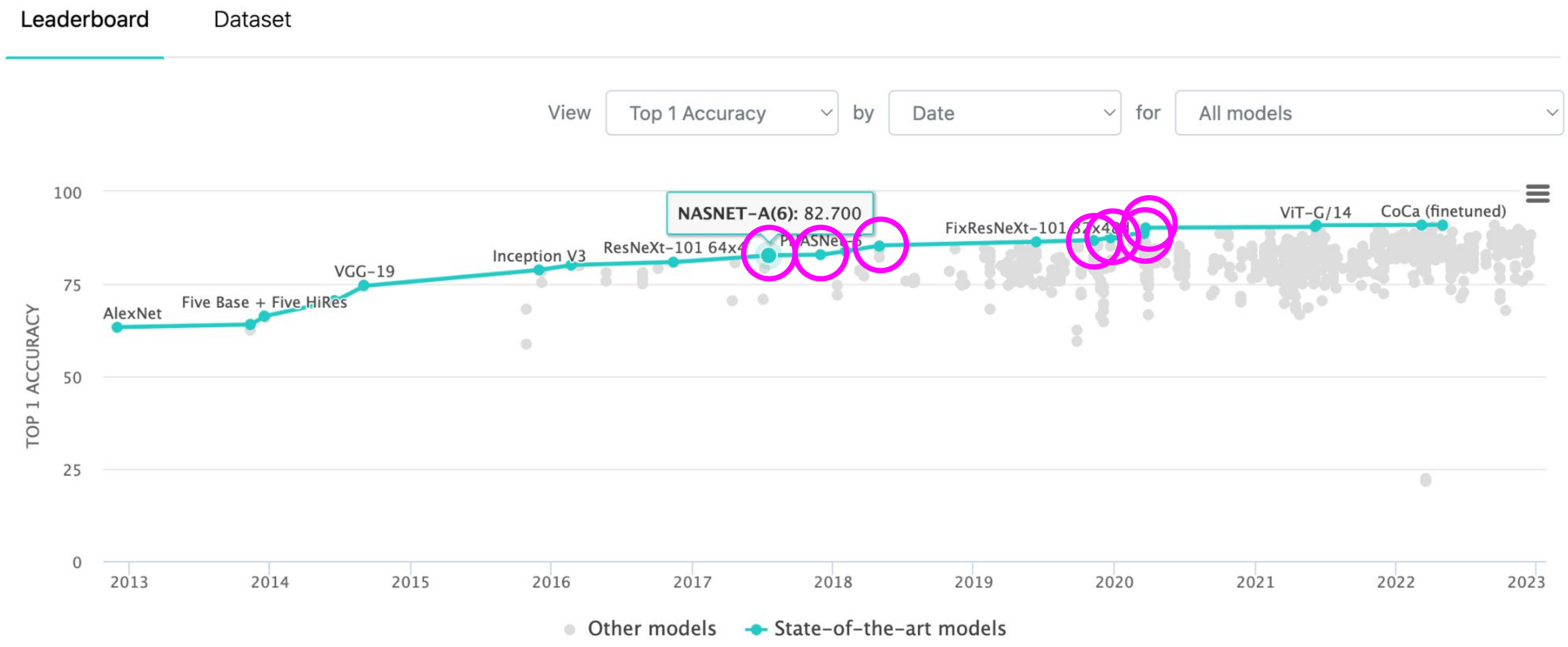


Stable Diffusion (2022)



AFNO (2022)

State of the Art on ImageNet



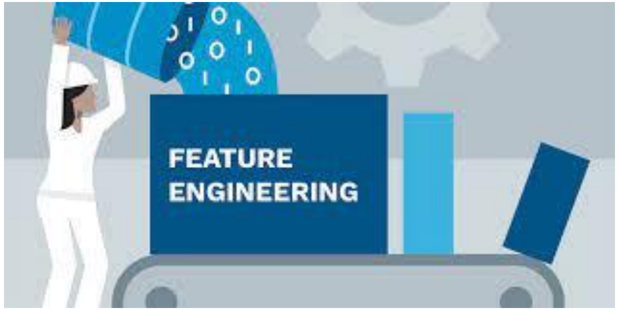
https://paperswithcode.com/sota/image-classification-on-imagenet

1950s

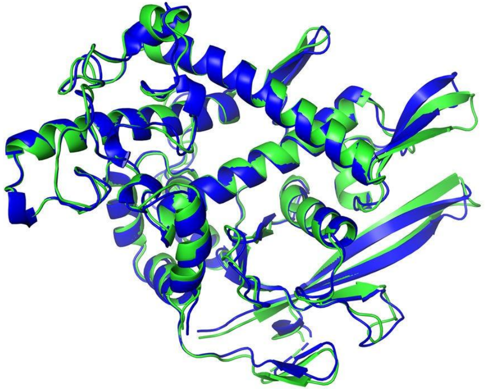
2012

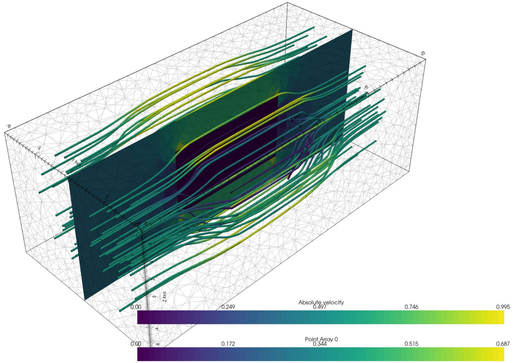
2017

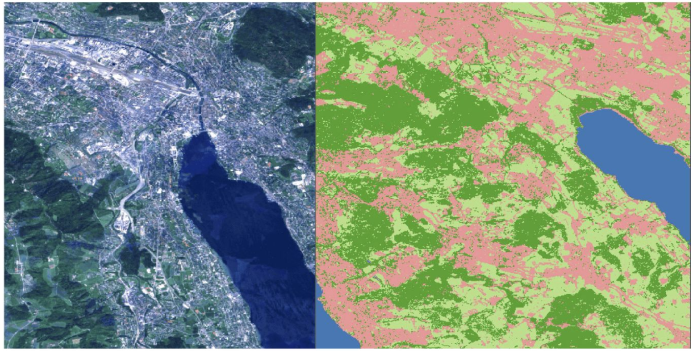
2023

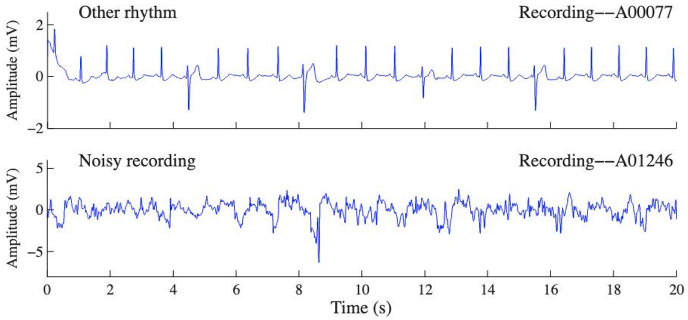


Diverse Tasks





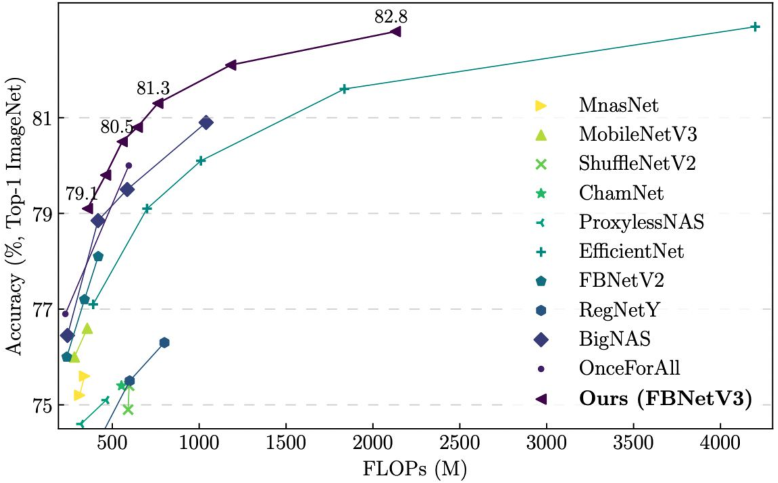


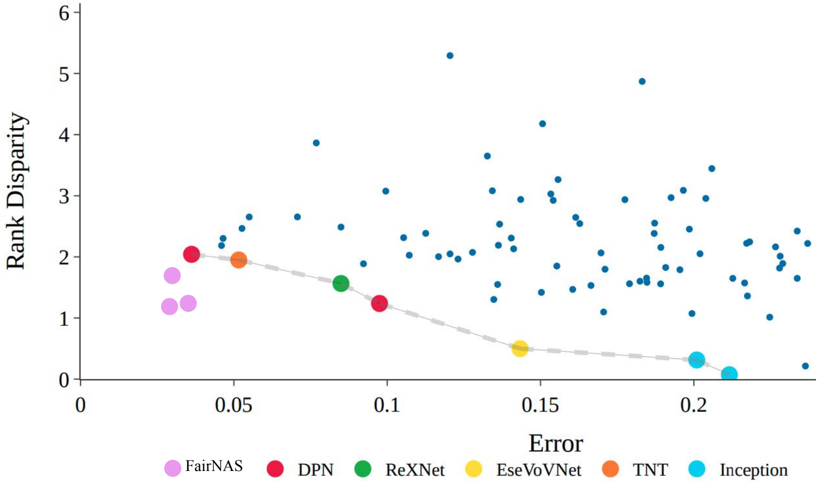


[TRKSST (2022)]

[TRPNJSRTNW (2022)]

Multi-Objective





DWZWHWCTYVG (2021)

SDDWHG (2022)

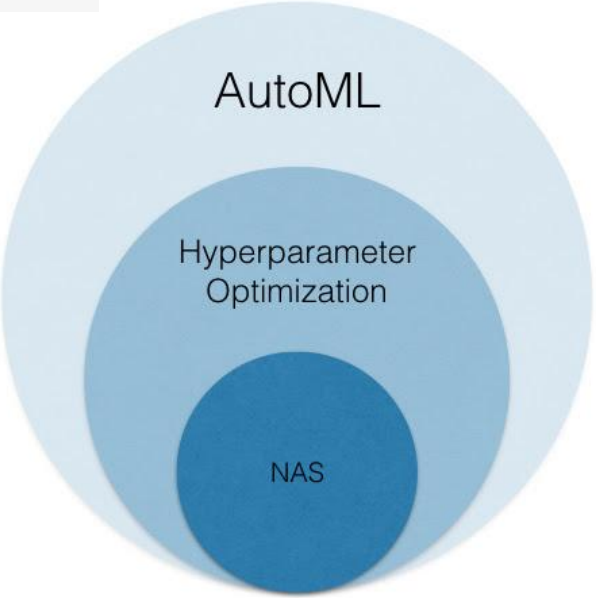
What is neural architecture search?

Neural architecture search (NAS) is a technique for automatically finding the best architecture for a neural network for a particular task. It involves using a search algorithm to explore the space of possible network architectures, and selecting the one that performs the best on a given dataset.

Given a search space *A*,

$$\min_{a \in \mathcal{A}} \quad \mathcal{L}_{\text{val}}\left(w^*(a), a\right)$$

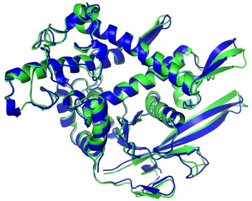
$$\text{s.t.} \quad w^*(a) = \operatorname{argmin}_w \; \mathcal{L}_{\text{train}}\left(w, a\right)$$



When to run NAS?

- Well-known datasets

- New datasets



- Constrained/multi-objective

Automate the design of **high-performing** models using **theoretical analyses**, **algorithm design**, and large-scale **empirical analyses**.

Neural architecture search

WNNS (NeurIPS 2020)
WNS (AAAI 2021)
WNS (UAI 2021)
Y*W*SH (NeurIPS 2021)
WZRLH (NeurIPS 2021)
M*W*ZKZMSYH (ICLR 2022)
WKTSBD ICLR-Blog (2022)
K*W*T*Z*SH (NeurIPS 2022)
PWJNIR (arXiv 2022)
WSSREZDH (arXiv 2023)

Hyperparameter optimization

BNVW (COLT 2017)
BDW (NeurIPS 2018)
MKVDW (NeurIPS 2022)

Bias & explainability

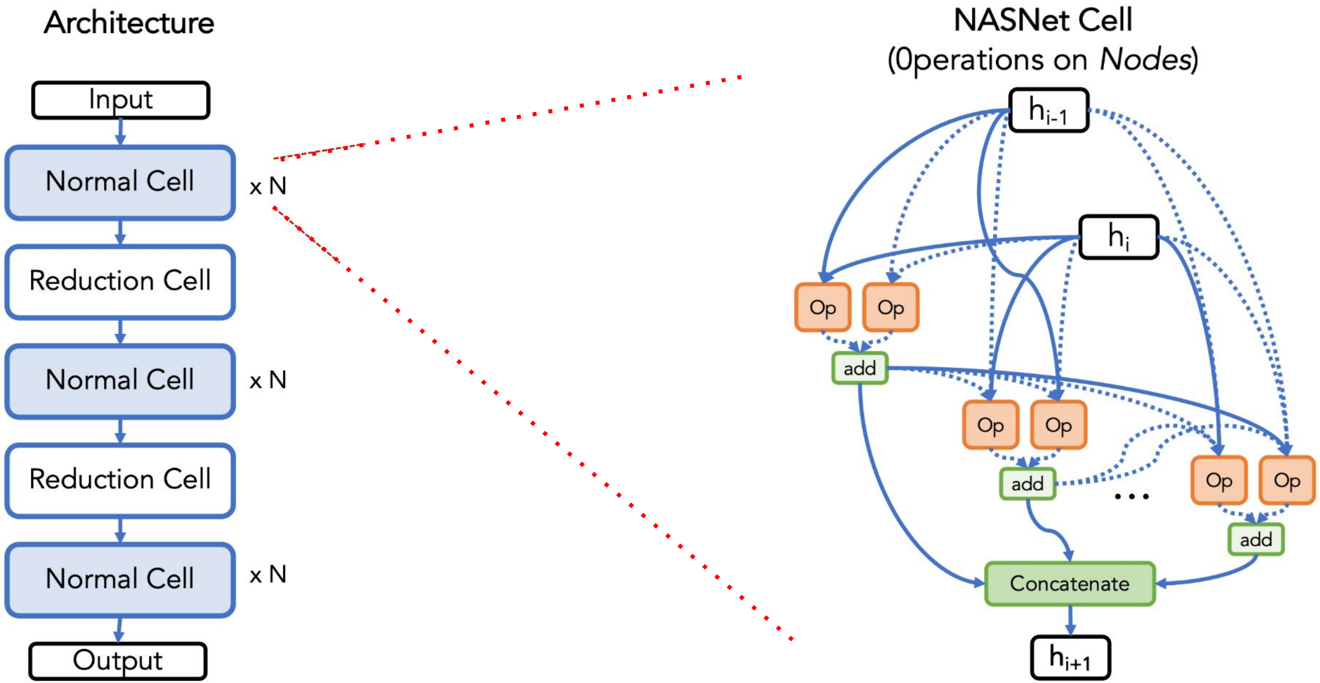
SWG (NeurIPS 2020)
LKWN (NeurIPS 2021)
SDDWHG (arXiv 2022)

Enabling good scientific practices

Outline

- Introduction
- **NAS**
  - **Algorithm design**
  - **Mitigating bias**
- HPO: theoretical results
- Future directions

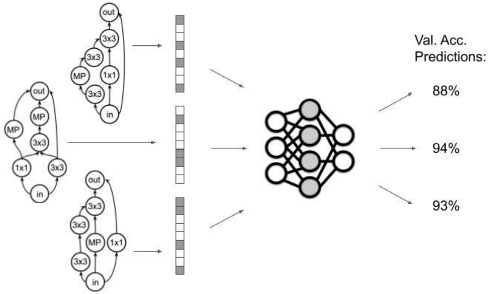
Search Spaces



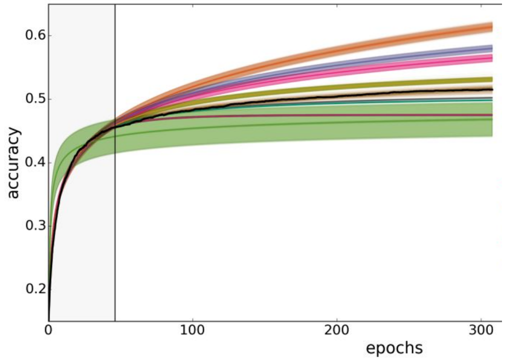
ZVSL (2018)

Performance Prediction

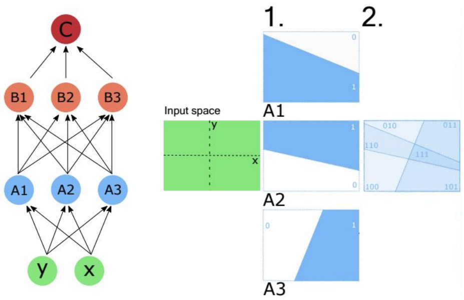
Predict the (relative) accuracy of an architecture, without fully training it



Model-based



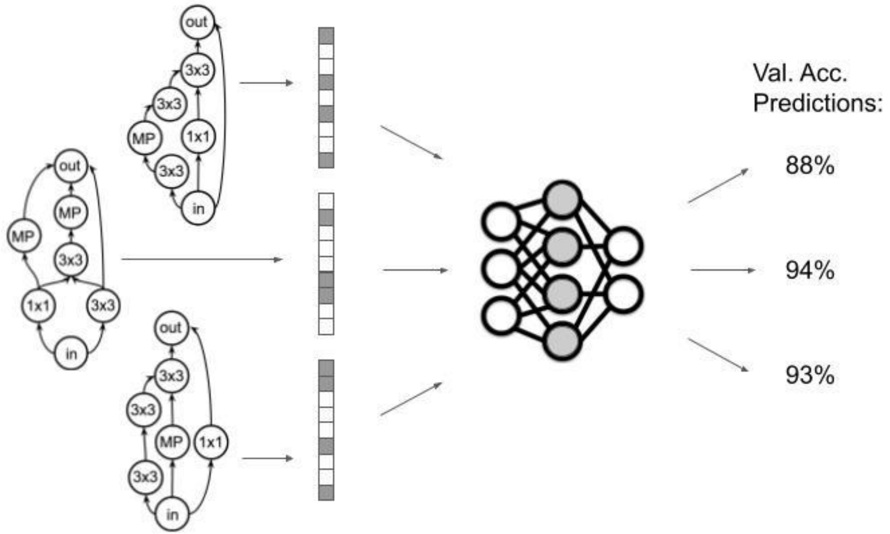
Learning curve extrapolation



Zero-cost proxies

Model-Based Predictors

Train a surrogate model



- Gaussian processes
  - KNSPX (2018)
  - JSH (2018)
- Boosted trees
  - LTWQCL (2020)
  - ZSZLKH (2020)
- GNNs
  - SPXLKZ (2019)
  - WLLCBK (2019)
- Specialized encodings
  - WNS (2019)
  - NZZWY (2020)

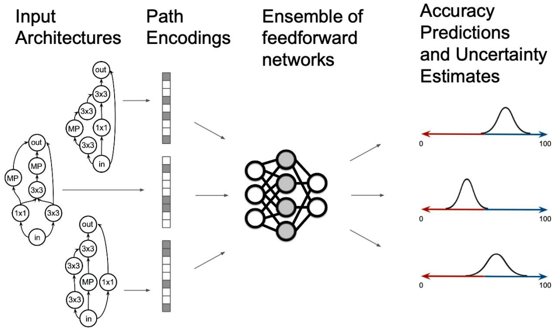
Bayesian Optimization

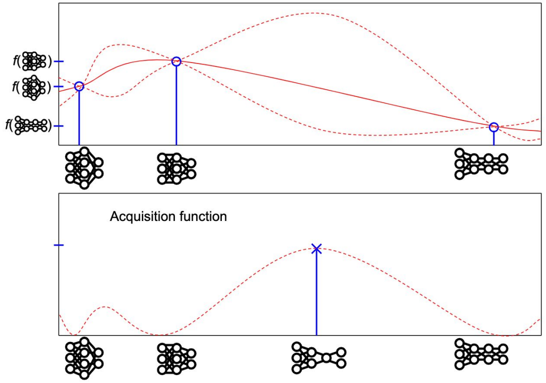
Algorithm 1 BANANAS

Input: Search space A, dataset D, parameters t_0, T, M, c, x, acquisition function ϕ, function $f(a)$ returning validation error of a after training.

1. Draw t_0 architectures $a_0, \ldots, a_{t_0}$ uniformly at random from A and train them on D.
2. For t from t_0 to T,
   i. Train an ensemble of meta neural networks on $\{(a_0, f(a_0)), \ldots, (a_t, f(a_t))\}$.
   ii. Generate a set of c candidate architectures from A by randomly mutating the x architectures a from $\{a_0, \ldots, a_t\}$ that have the lowest value of $f(a)$.
   iii. For each candidate architecture a, evaluate the acquisition function $\phi(a)$.
   iv. Denote a_{t+1} as the candidate architecture with minimum $\phi(a)$, and evaluate $f(a_{t+1})$.

Output: $a^* = \text{argmin}_{t=0,\ldots,T} f(a_t)$.





WNS (2019), SPXLKZ (2019), MCY (2019)

Bayesian Optimization

Algorithm 1 BANANAS

Input: Search space A, dataset D, parameters t_0, T, M, c, x, acquisition function ϕ, function $f(a)$ returning validation error of a after training.

1. Draw t_0 architectures $a_0, \ldots, a_{t_0}$ uniformly at random from A and train them on D.
2. For t from t_0 to T,
   i. Train an ensemble of meta neural networks on $\{(a_0, f(a_0)), \ldots, (a_t, f(a_t))\}$.
   ii. Generate a set of c candidate architectures from A by randomly mutating the x architectures a from $\{a_0, \ldots, a_t\}$ that have the lowest value of $f(a)$.
   iii. For each candidate architecture a, evaluate the acquisition function $\phi(a)$.
   iv. Denote a_{t+1} as the candidate architecture with minimum $\phi(a)$, and evaluate $f(a_{t+1})$.

Output: $a^* = \text{argmin}_{t=0,\ldots,T} f(a_t)$.

WNS (AAAI 2021)

- Architecture encoding
- Uncertainty calibration
- Neural predictor architecture
- Acquisition optimization strategy
- Acquisition function

BANANAS

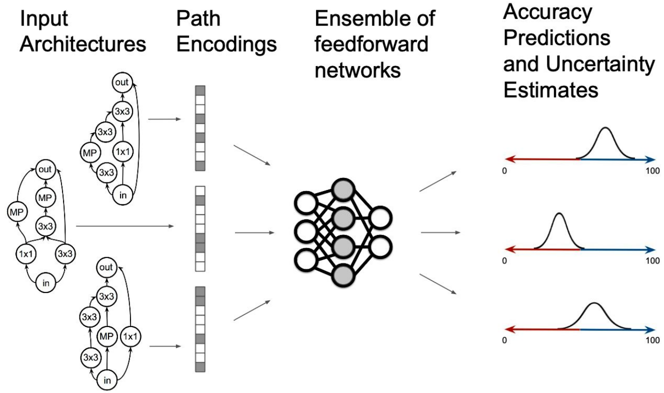
Algorithm 1 BANANAS

Input: Search space A, dataset D, parameters t_0, T, M, c, x, acquisition function ϕ, function $f(a)$ returning validation error of a after training.

1. Draw t_0 architectures $a_0, \ldots, a_{t_0}$ uniformly at random from A and train them on D.
2. For t from t_0 to T,
   i. Train an ensemble of meta neural networks on $\{(a_0, f(a_0)), \ldots, (a_t, f(a_t))\}$.
   ii. Generate a set of c candidate architectures from A by randomly mutating the x architectures a from $\{a_0, \ldots, a_t\}$ that have the lowest value of $f(a)$.
   iii. For each candidate architecture a, evaluate the acquisition function $\phi(a)$.
   iv. Denote a_{t+1} as the candidate architecture with minimum $\phi(a)$, and evaluate $f(a_{t+1})$.

Output: $a^* = \text{argmin}_{t=0,\ldots,T} f(a_t)$.

WNS (AAAI 2021)



Path encoding, ensemble

Small mutations

Independent Thompson Sampling

Follow-up work

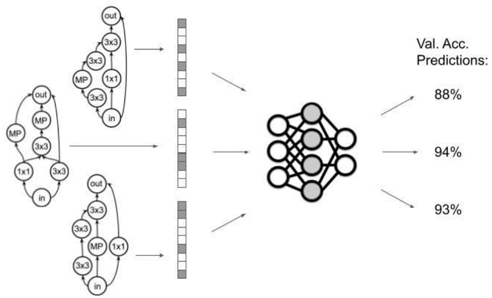
- Improved surrogate (WNTWHL, 2020)
- Multi-objective; NAS+HPO (GHIMSBEDLH, 2021)
- Improved acq. fn. opt. (SPBB, 2021)
- Adding LCE (Y*W*SH, 2021)
- Adding ZC Proxies (WZRLH, 2021)
- Adding ZC Proxies (K*W*Z*T*SH, 2022)

Subsequent benchmark

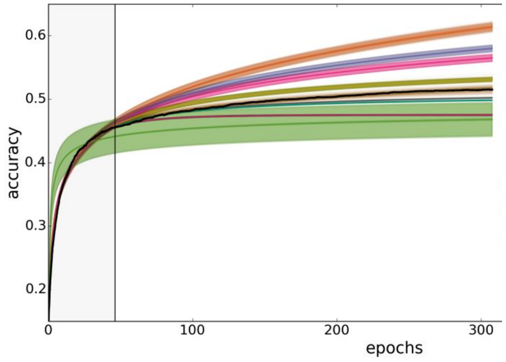
- NAS-Bench-301 (ZSZLKH, 2020)

Performance Prediction

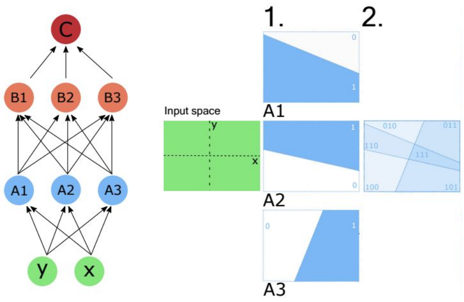
Predict the (relative) accuracy of an architecture, without fully training it



Model-based



Learning curve extrapolation



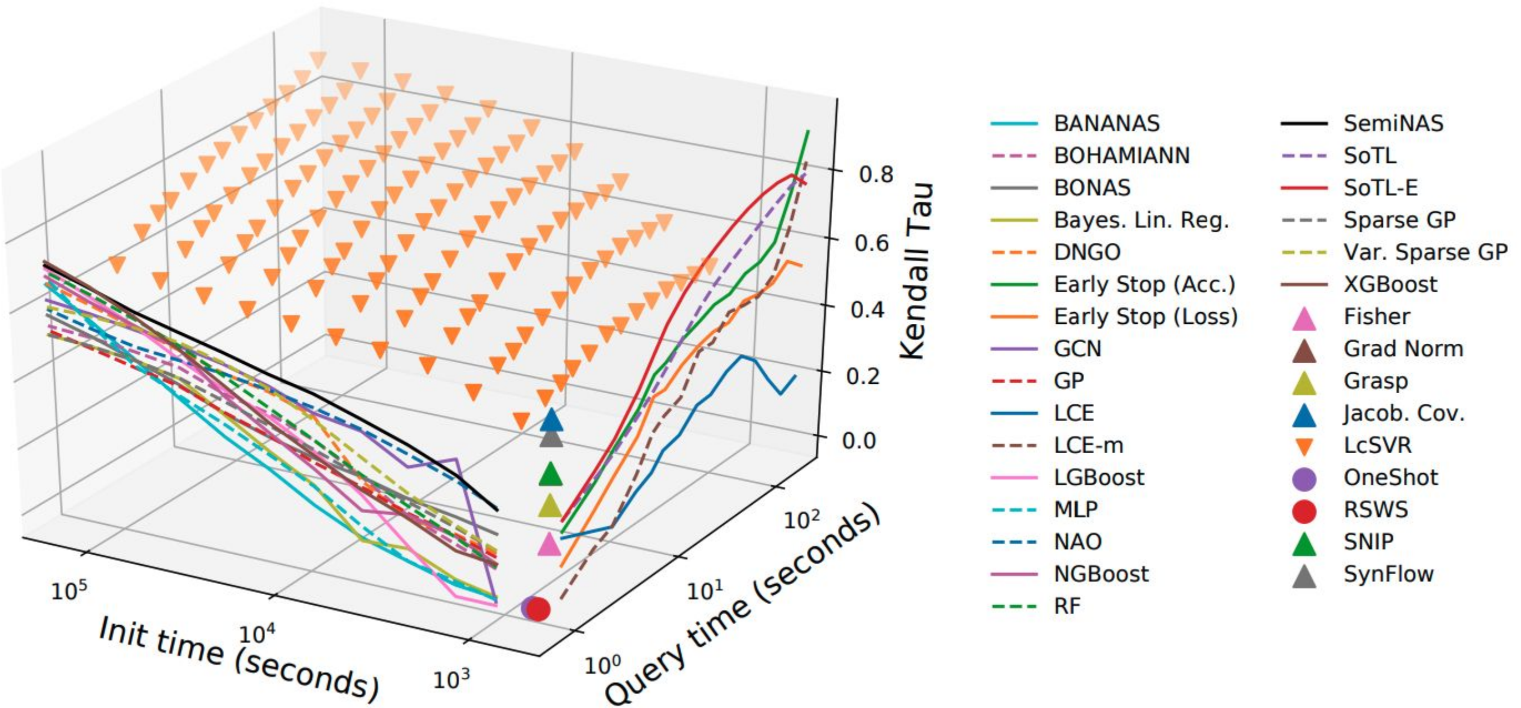
Zero-cost proxies

WNNS (NeurIPS 2020)
WNS (AAAI 2021)

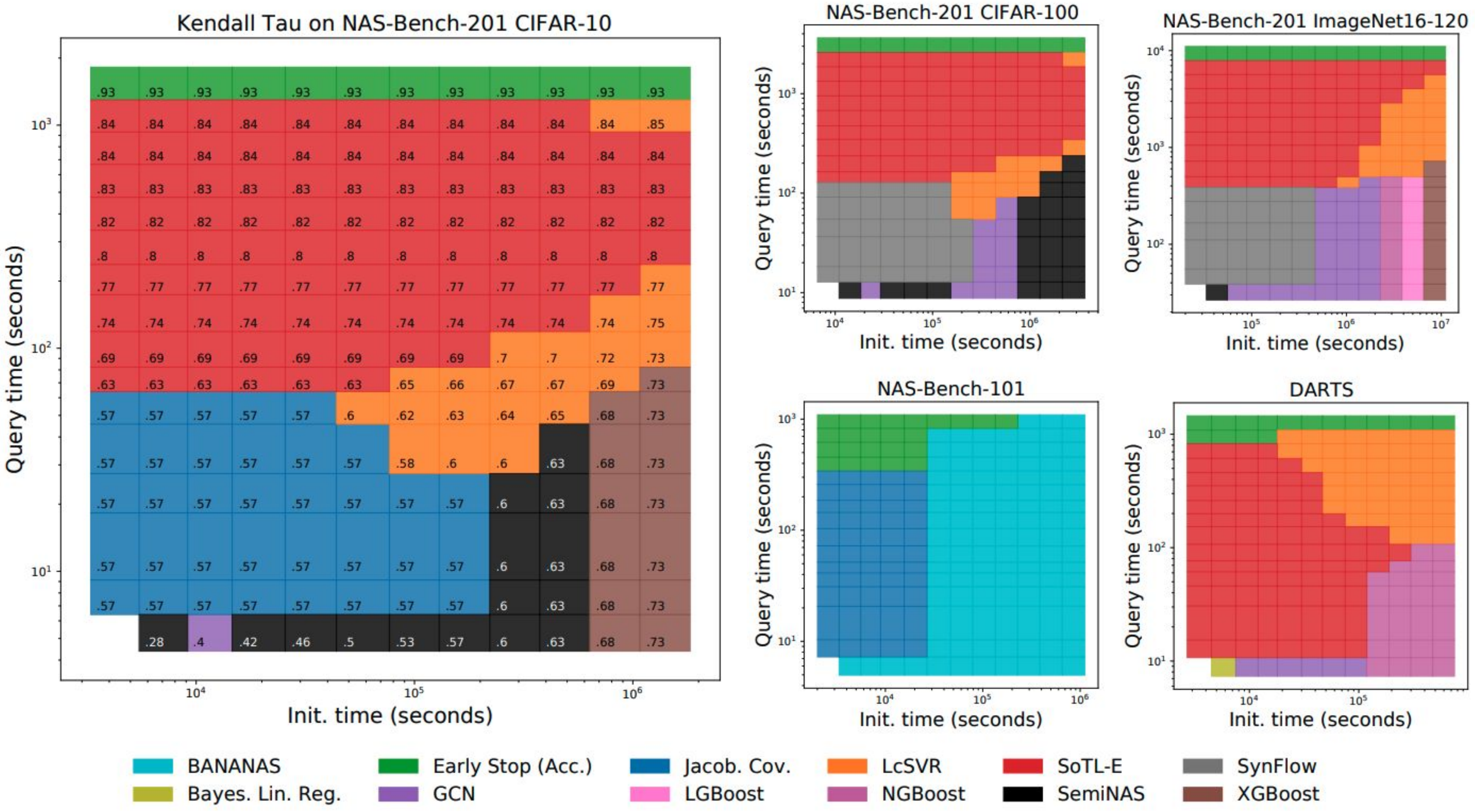
WZRLH (NeurIPS 2021)
Y*W*SH (NeurIPS 2021)

WKTSBD (ICLR-Blog 2022)
K*W*T*Z*SH (NeurIPS 2022)

Kendall Tau on NAS-Bench-201 CIFAR-10



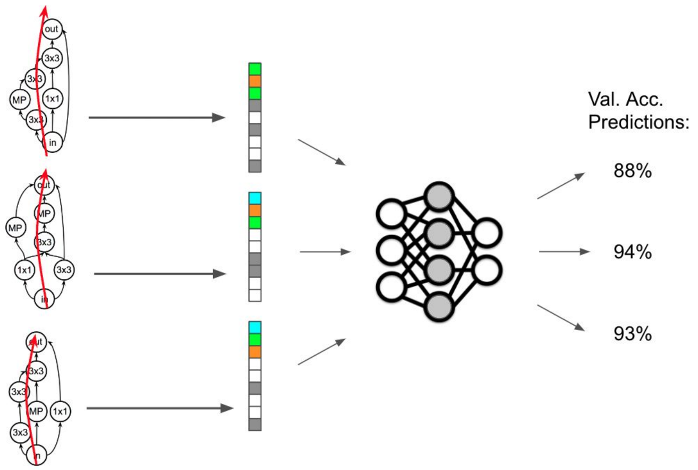
WZRLH (NeurIPS 2021)

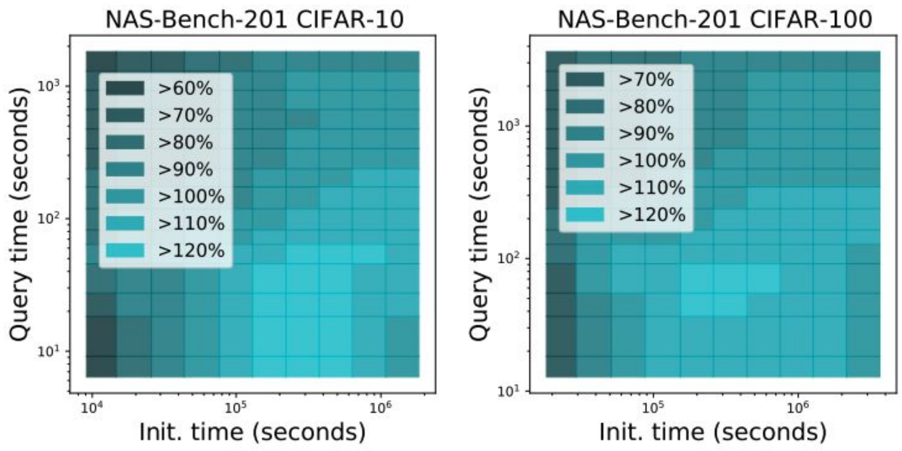


WZRLH (NeurIPS 2021)

The Omnipotent Predictor

Combine best predictors from each family





WZRLH (NeurIPS 2021)

How do we set up
neural architecture search
for a new application?

Face Recognition

1. **For one-to-one matching, the team saw higher rates of false positives for Asian and African American faces relative to images of Caucasians.** The differentials often ranged from a factor of $\boxed{\text{10 to 100 times,}}$ depending on the individual algorithm. False positives might present a security concern to the system owner, as they may allow access to impostors.
2. **Among U.S.-developed algorithms, there were similar high rates of false positives in one-to-one matching for Asians, African Americans and native groups** (which include Native American, American Indian, Alaskan Indian and Pacific Islanders). The American Indian demographic had the highest rates of false positives.

'The Computer Got It Wrong': How Facial Recognition Led To False Arrest Of Black Man

June 24, 2020 · 8:00 AM ET

BOBBY ALLYN

A US government study confirms most face recognition systems are racist

by Karen Hao December 20, 2019

A U.S. Customs and Border Protection officer helps a passenger navigate a facial recognition kiosk at the airport.
DAVID J. PHILLIP/AP

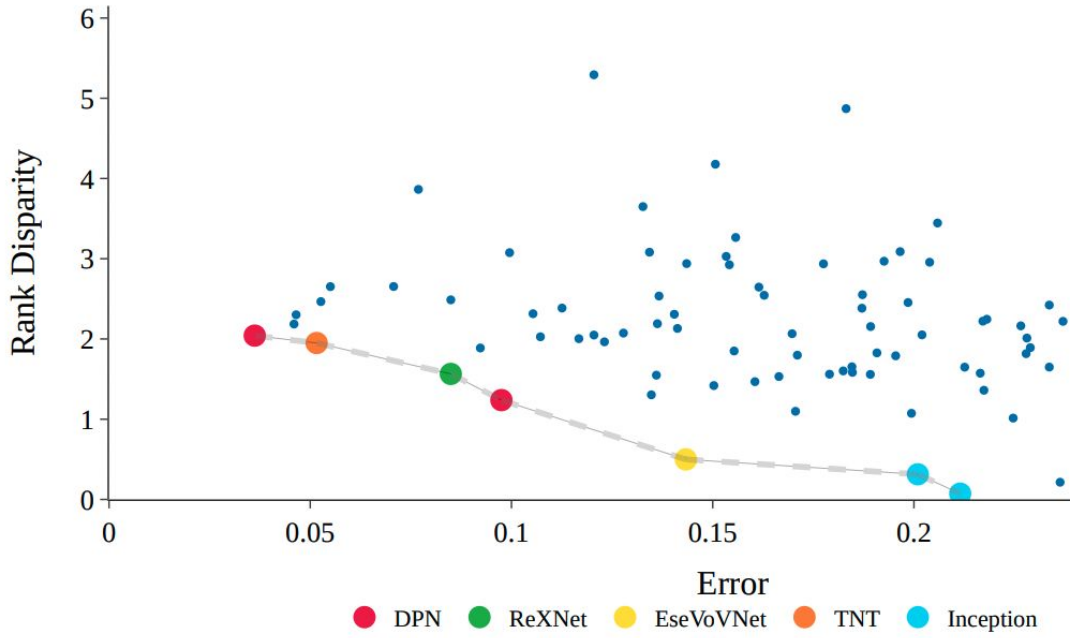
https://www.nist.gov/news-events/news/2019/12/nist-study-evaluates-effects-race-age-sex-face-recognition-software

Architectures

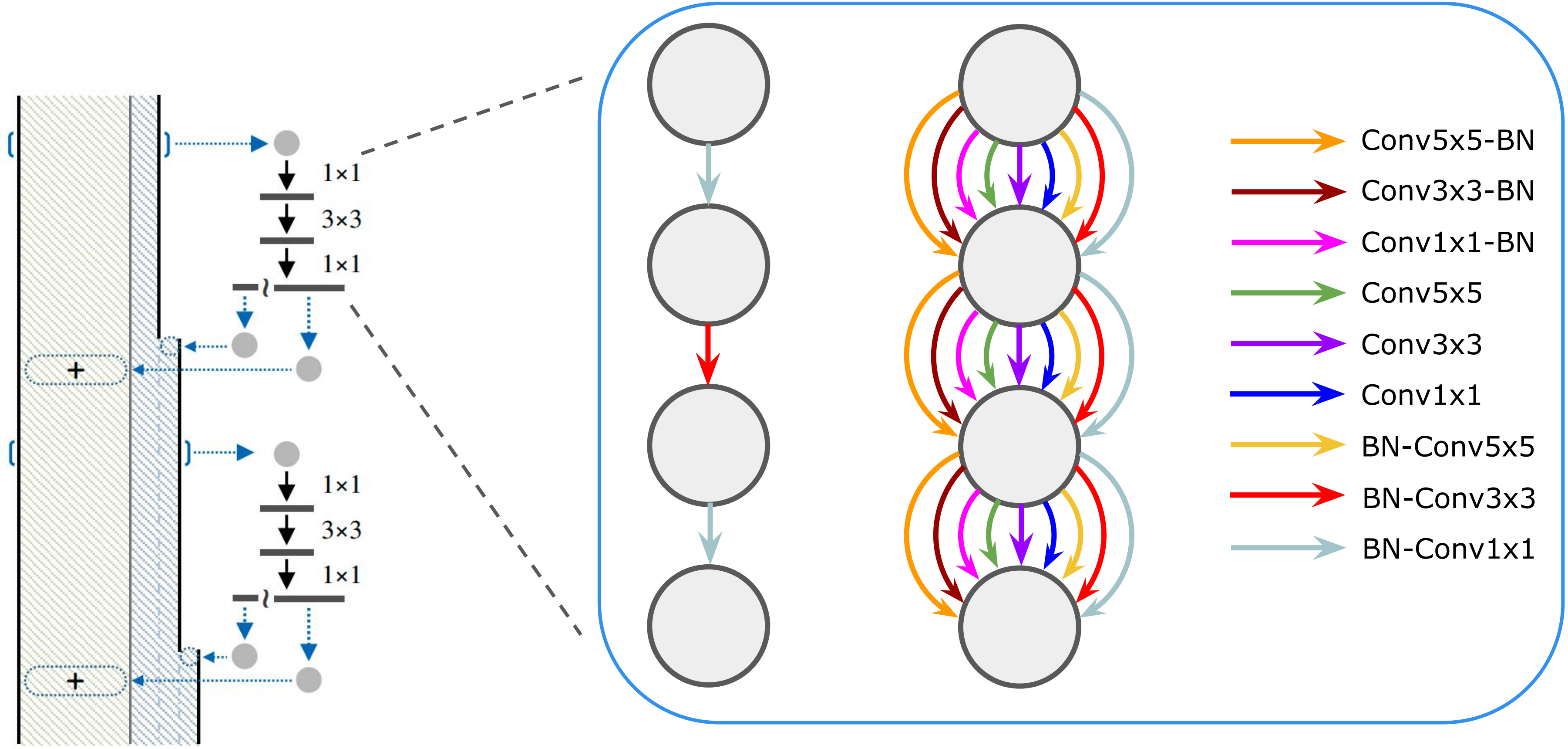
Examples:
Inception
ViTransformer
VovNet
MobileNetV3
Dual Path Net

Hyperparams

Examples:
MagFace (head)
ArcFace (head)
CosFace (head)
Optimizer
Learning Rate

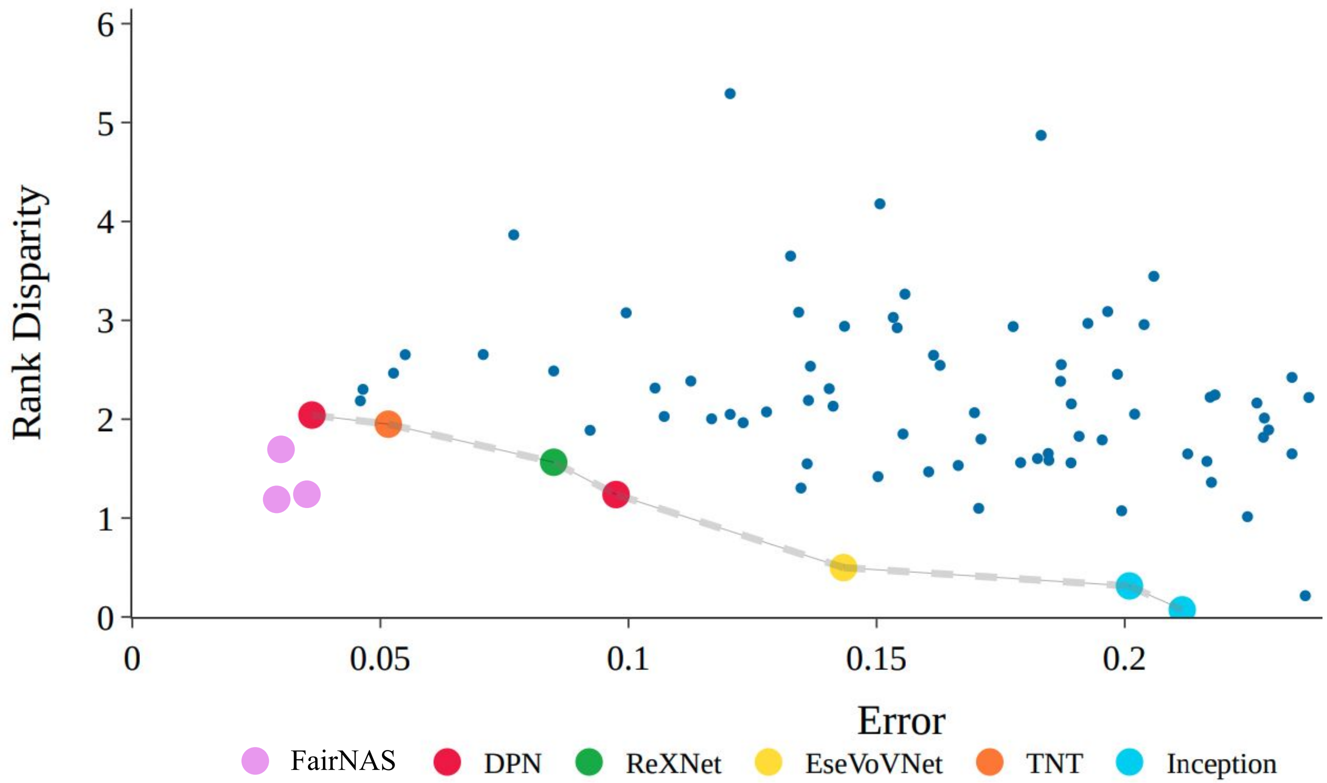


SDDWHG (arXiv 2022)



DPN (2018)

SDDWHG (arXiv 2022)



SDDWHG (arXiv 2022)

Outline

- Introduction
- NAS: algorithm design
- **HPO: theoretical results**
- Future directions

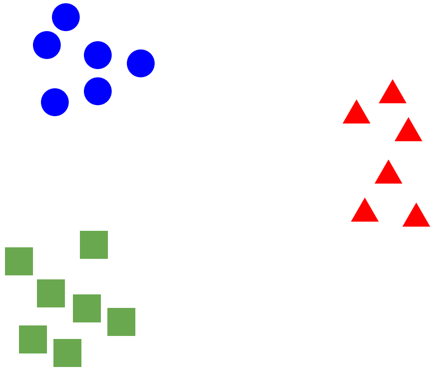
Data-Driven Algorithm Design

Optimization for **one** dataset → Optimization for a **distribution** of datasets

Supervised learning → **Unsupervised** learning (clustering)

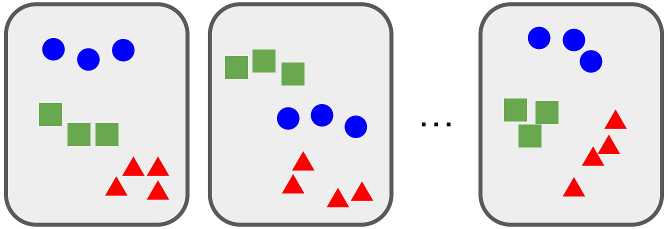
Clustering

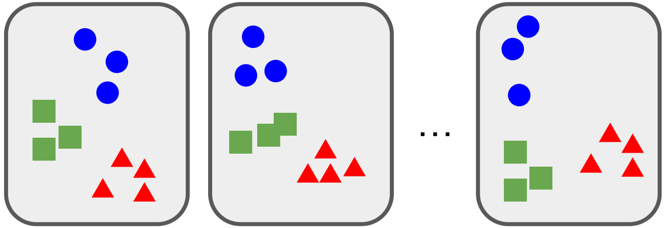
Input: dataset, k



- Find k clusters
- Minimize distance to ground-truth clustering

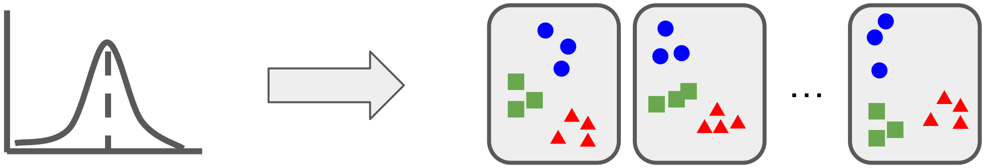
Data-Driven Algorithm Design





Data-Driven Algorithm Design

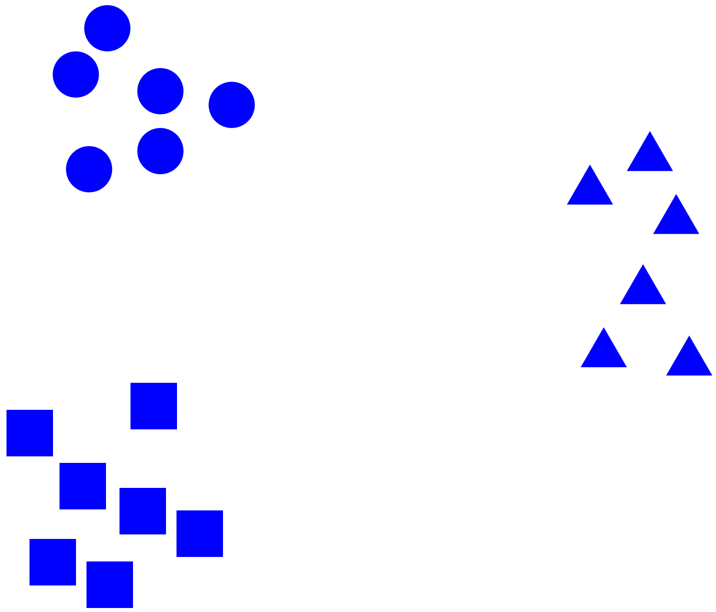
- Fix a parameterized clustering algorithm
- Receive training set of "typical" clustering instances



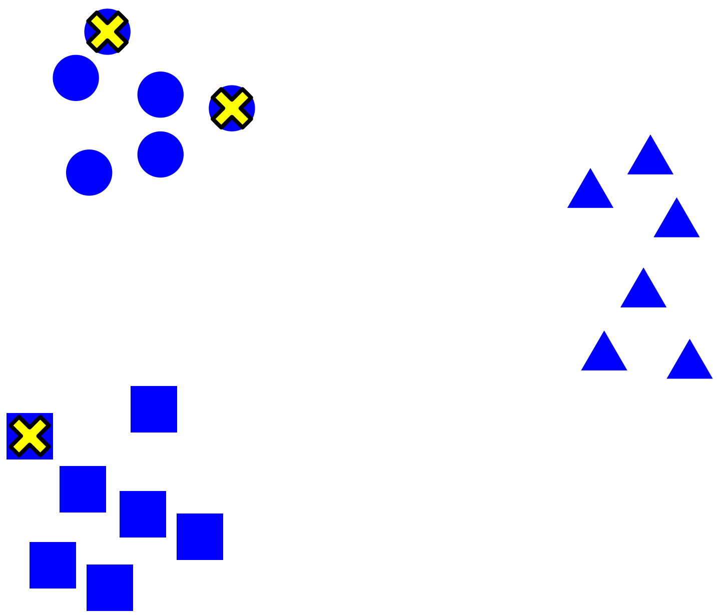
- Find hyperparameters with good average performance

How to find high-performing hyperparameters?
Will these hyperparameters have strong future performance?

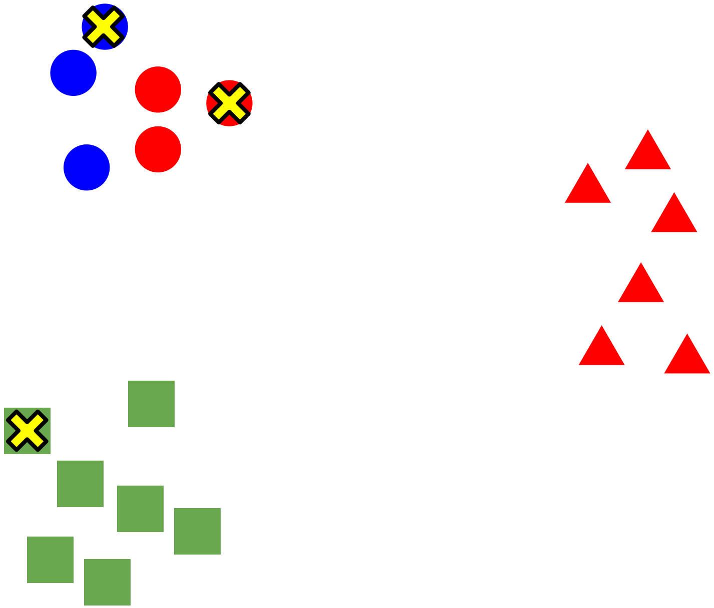
k-means



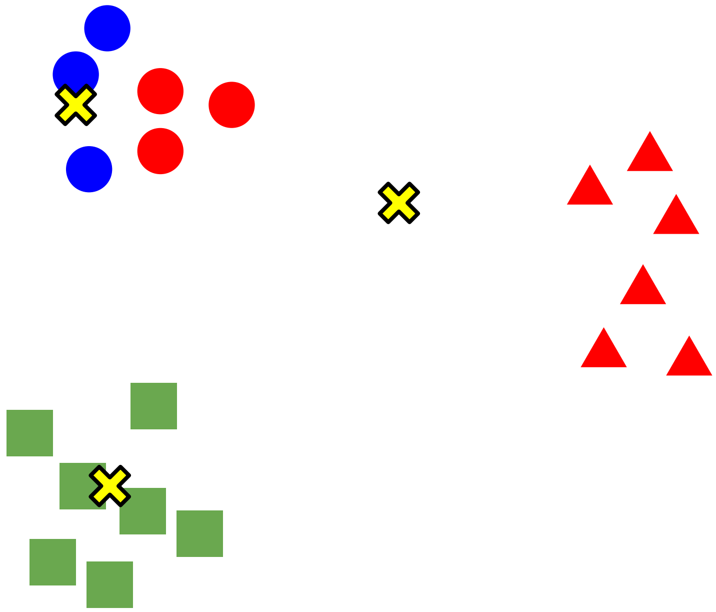
k-means



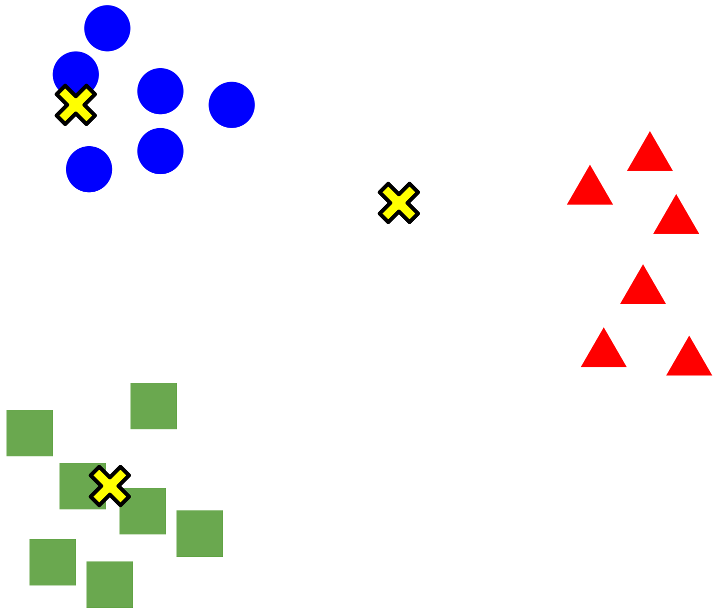
k-means



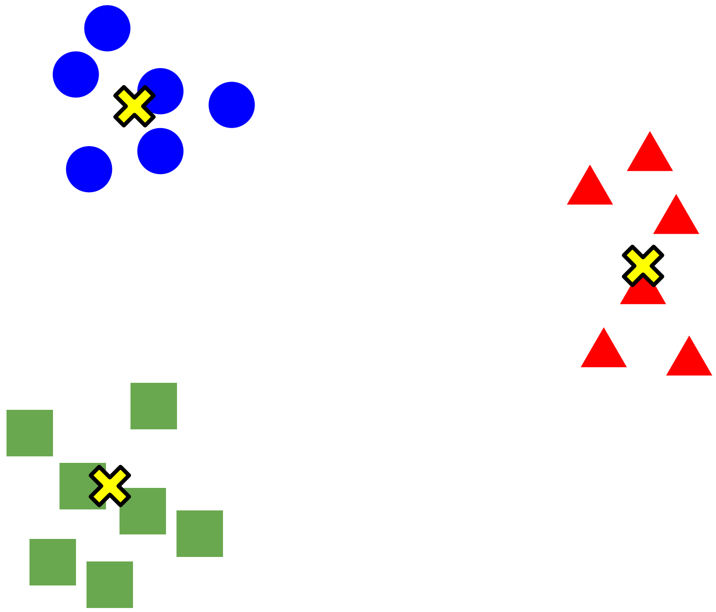
k-means



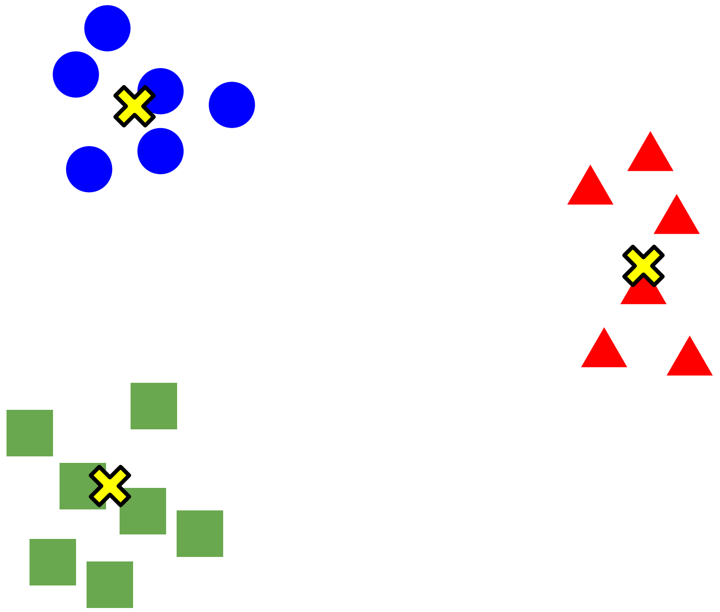
k-means



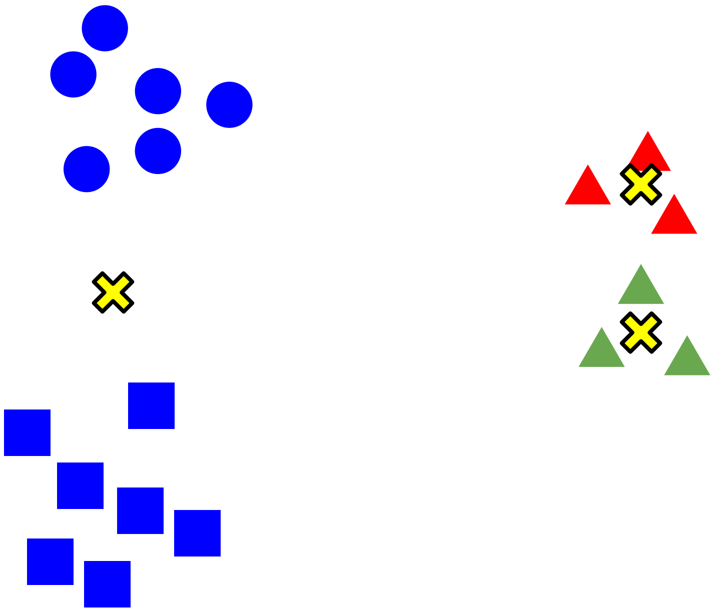
k-means



k-means

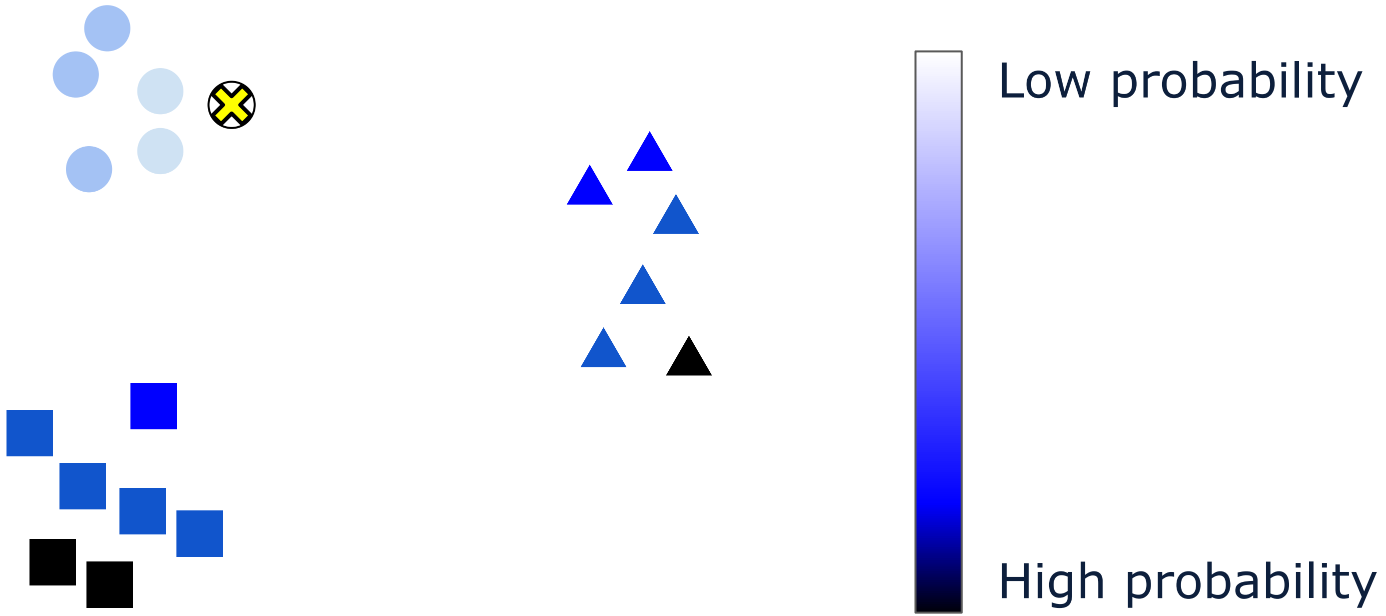


Initial centers are important!



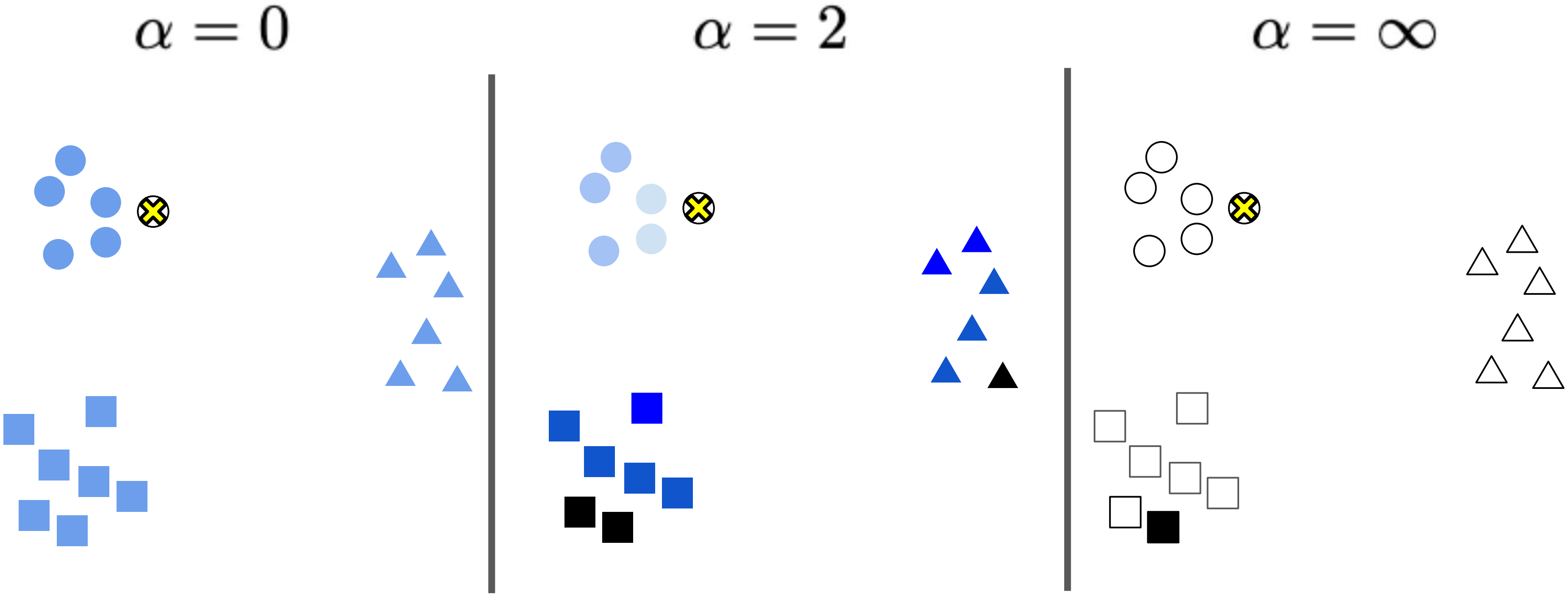
k-means++

d^2 sampling



A new family of clustering algorithms

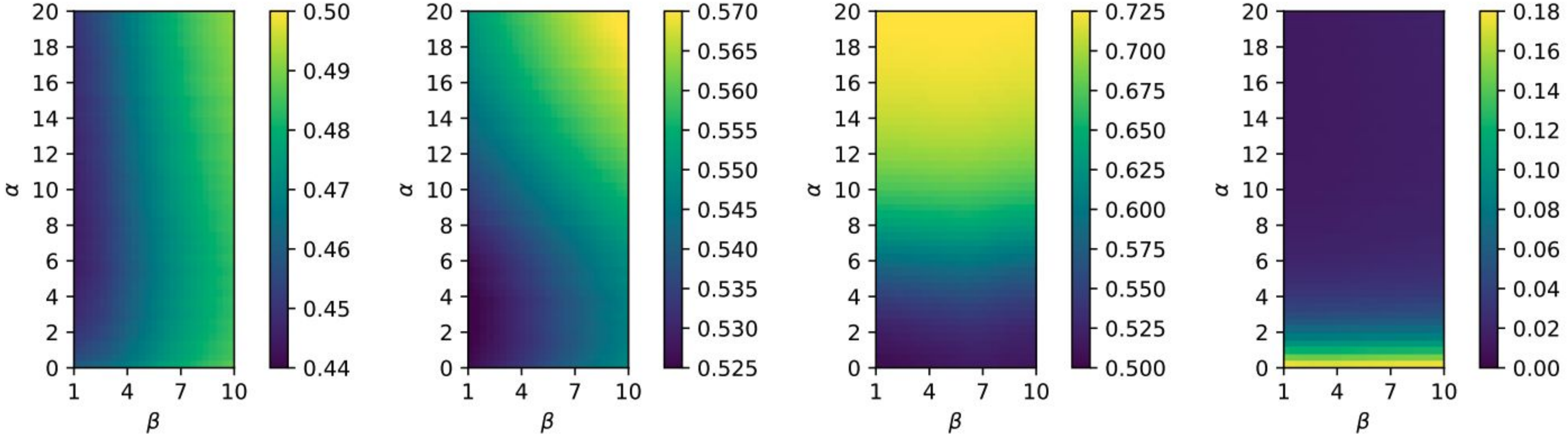
d^{α} sampling β-local search



BDW (NeurIPS 2018)

A new family of clustering algorithms

d^{α} sampling β-local search



(a) MNIST (b) CIFAR-10 (c) CNAE-9 (d) Gaussian Grid

BDW (NeurIPS 2018)

Theorem: Given $\tilde{O}\left(\frac{k \log n}{\epsilon^2}\right)$ sampled clustering instances, with high probability for all α, β, |**Avg** performance over training set - **expected** performance| < ε.

Key intuition:

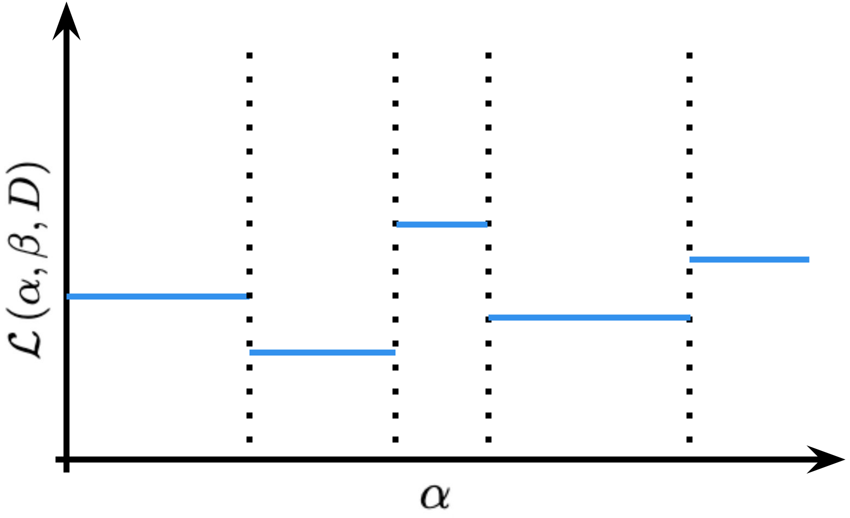
A set of hyperparameters with high performance **on the training set**, also achieves high performance **in the future**

BDW (NeurIPS 2018)

Theorem: Given $\tilde{O}\left(\frac{k \log n}{\epsilon^2}\right)$ sampled clustering instances, with high probability for all α, β, |**Avg** performance over training set - **expected** performance| < ε.

Key insight:

Bound the expected number of **discontinuities** of $\mathcal{L}(\alpha, \beta, D)$ as a function of α, β

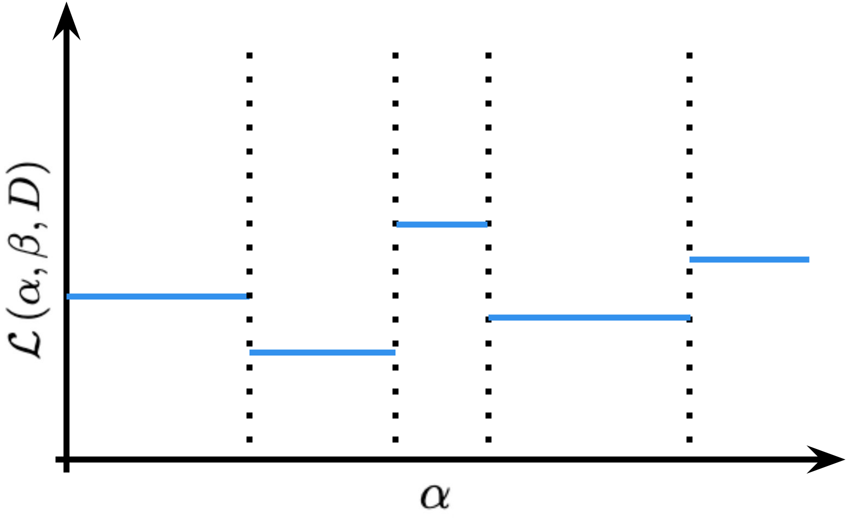


BDW (NeurIPS 2018)

Theorem: Given $\tilde{O}\left(\frac{k \log n}{\epsilon^2}\right)$ sampled clustering instances, with high probability for all α, β, |**Avg** performance over training set - **expected** performance| < ε.

Efficient algorithm:

Solve for the **discontinuities**



BDW (NeurIPS 2018)

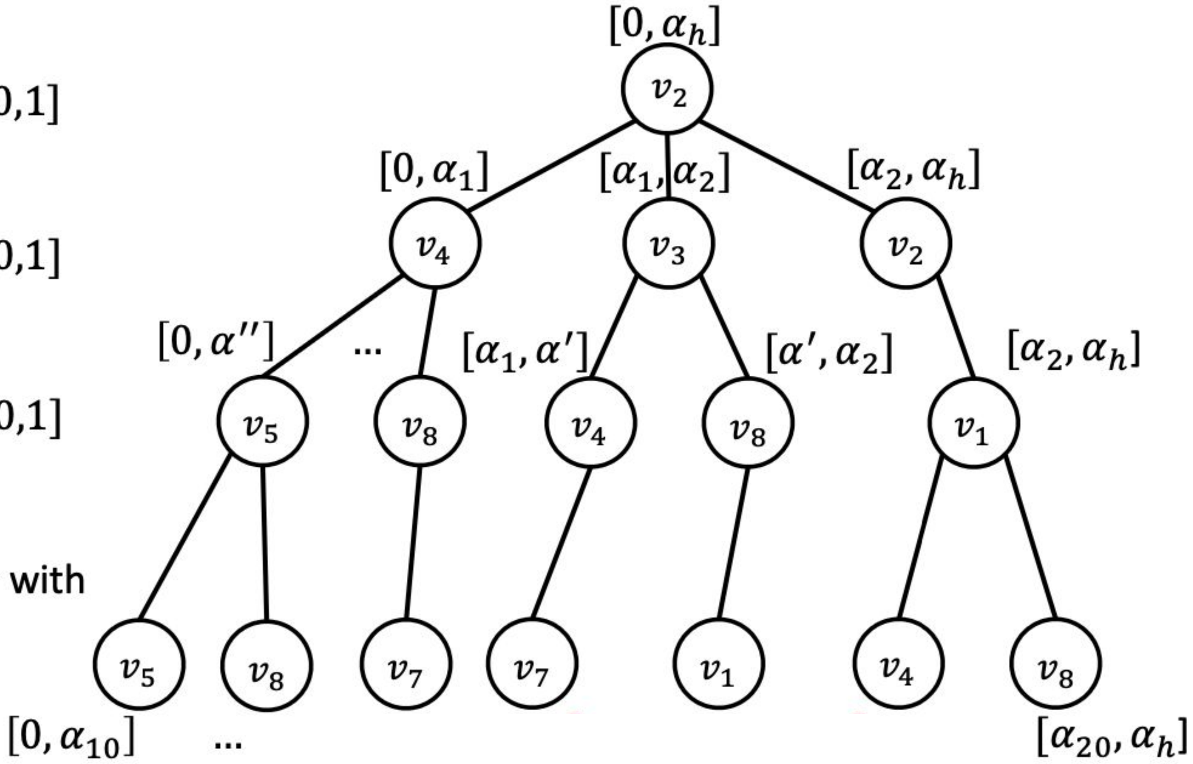
Execution Tree

Draw $z_1 \in [0,1]$

Draw $z_2 \in [0,1]$

Draw $z_3 \in [0,1]$

Full partition of $[0, \alpha_h]$ together with set of centers

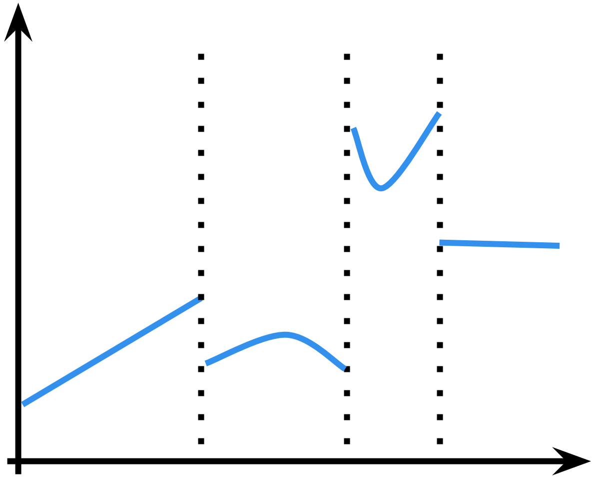


Generalizations

- Piecewise polynomial (BDDKSV, 2019)
- Approx. piecewise polynomial (BSV, 2020)

Beyond clustering

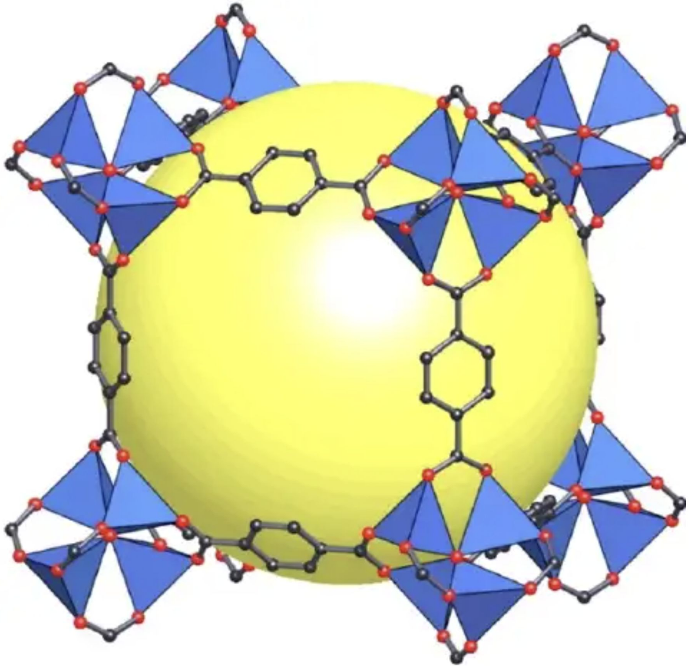
- Integer Programming (BDV, 2018)
- ElasticNet (BKST, 2022)

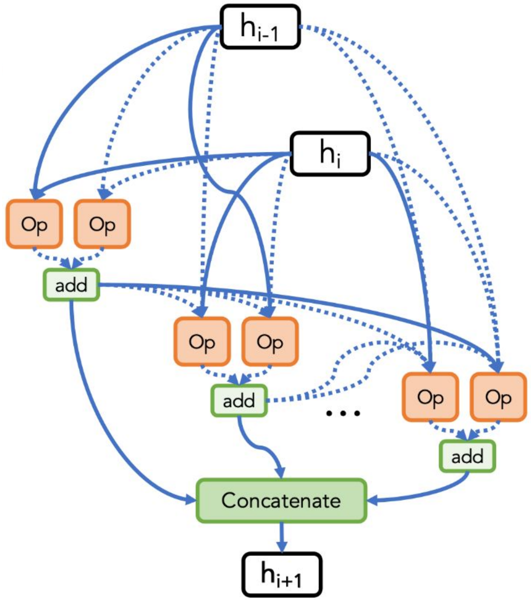


Outline

- Introduction
- NAS: algorithm design
- HPO: theoretical results
- **Future directions**
  - **Reticular chemistry**
  - **Climate sciences**

Tackling reticular chemistry with **AutoML**

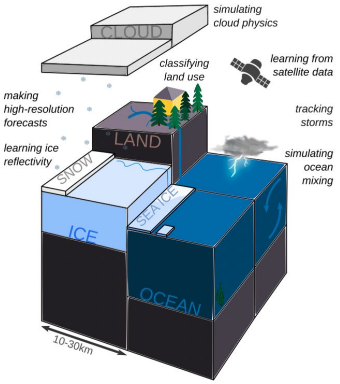


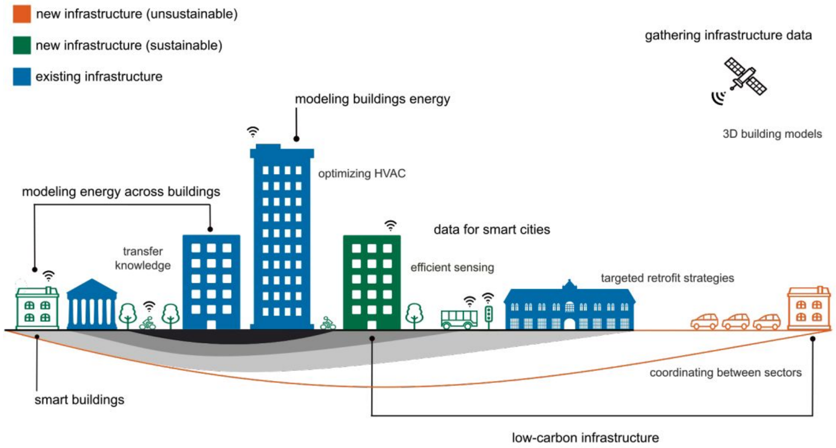


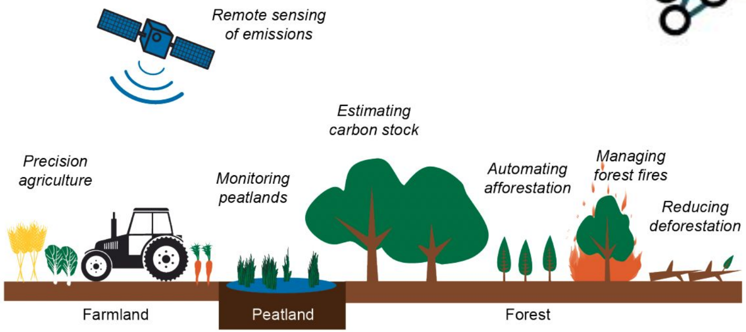
Kiyasseh et al. (2022)

Graph-like structures; uncertainty quantification

Tackling Climate Sciences with **AutoML**



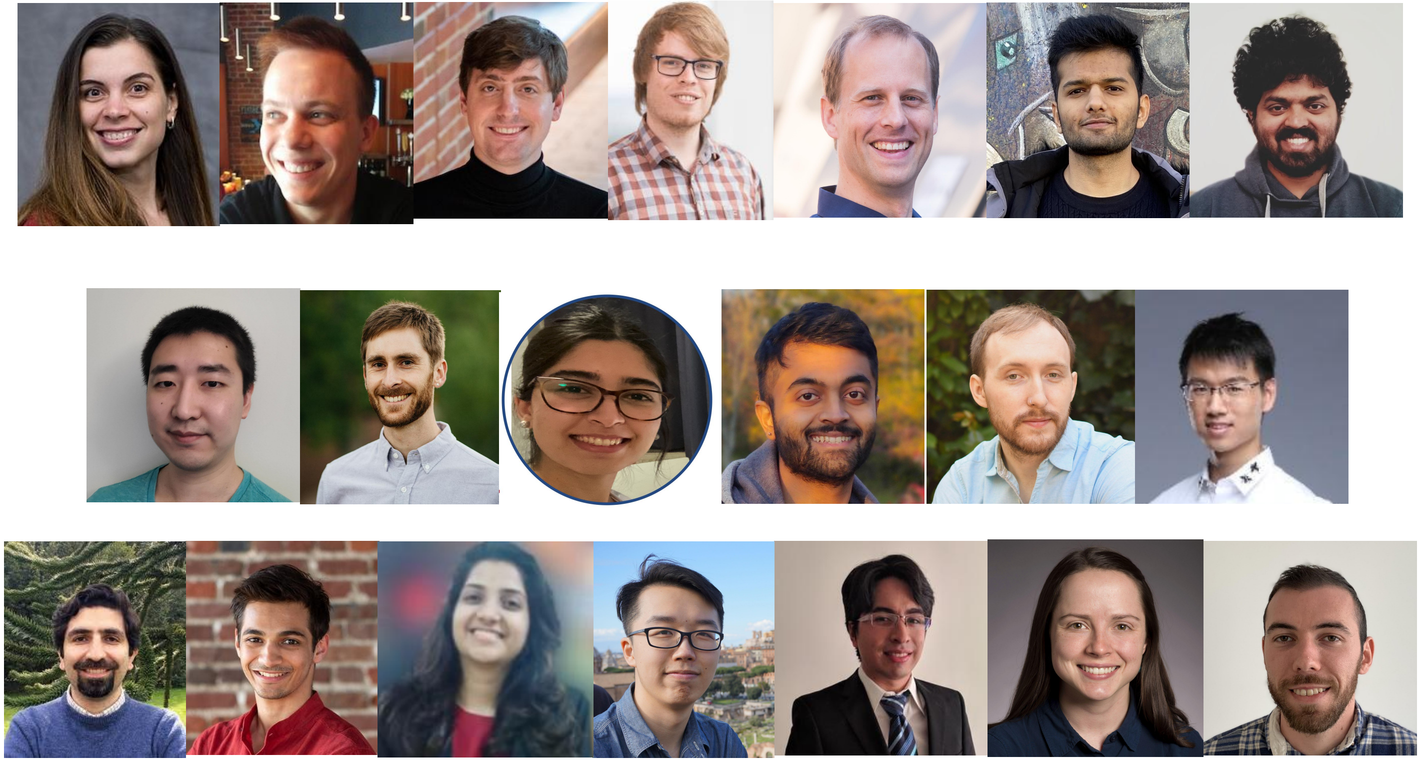




[Rolnick et al. (2018)]

[TRPNJSRTNW (2022)]

Thanks!



Slides (with hyperlinks): https://crwhite.ml/