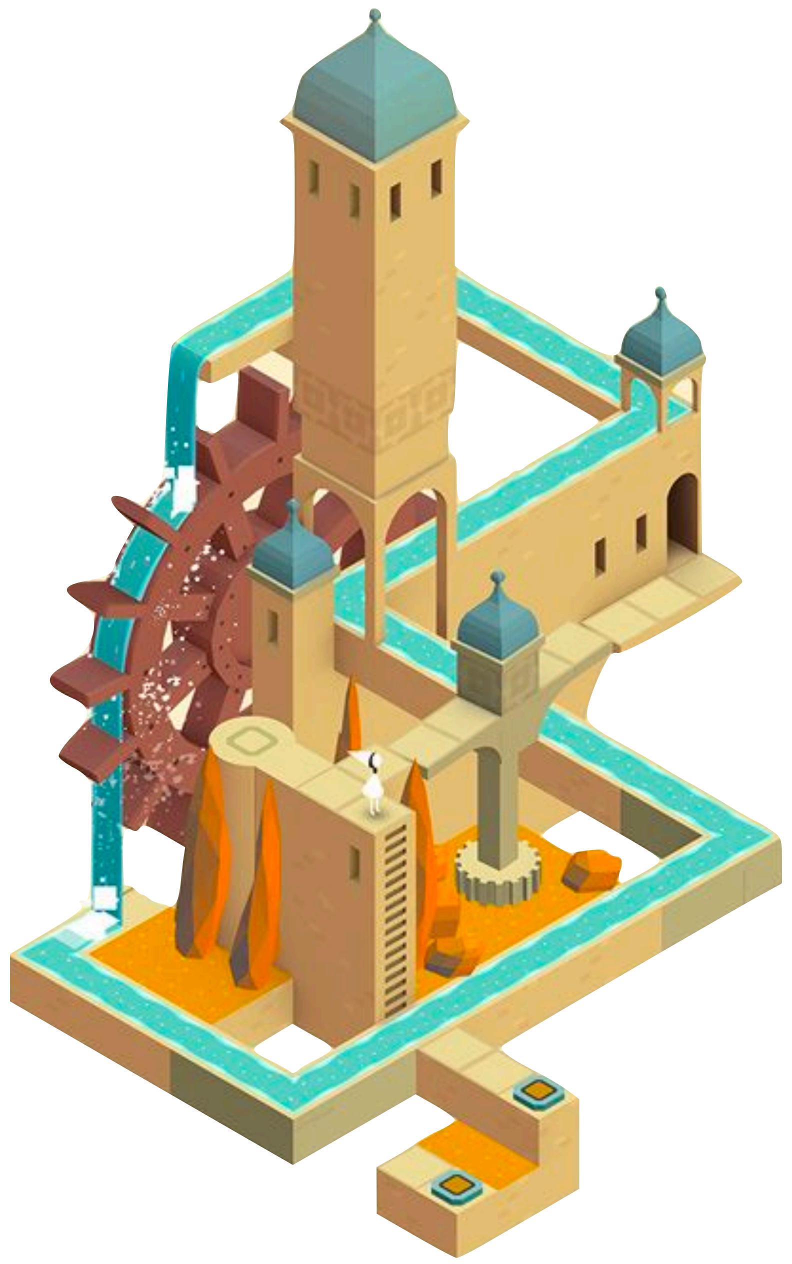


Learning how to learn with OpenML

On the road to self-learning automated machine learning systems

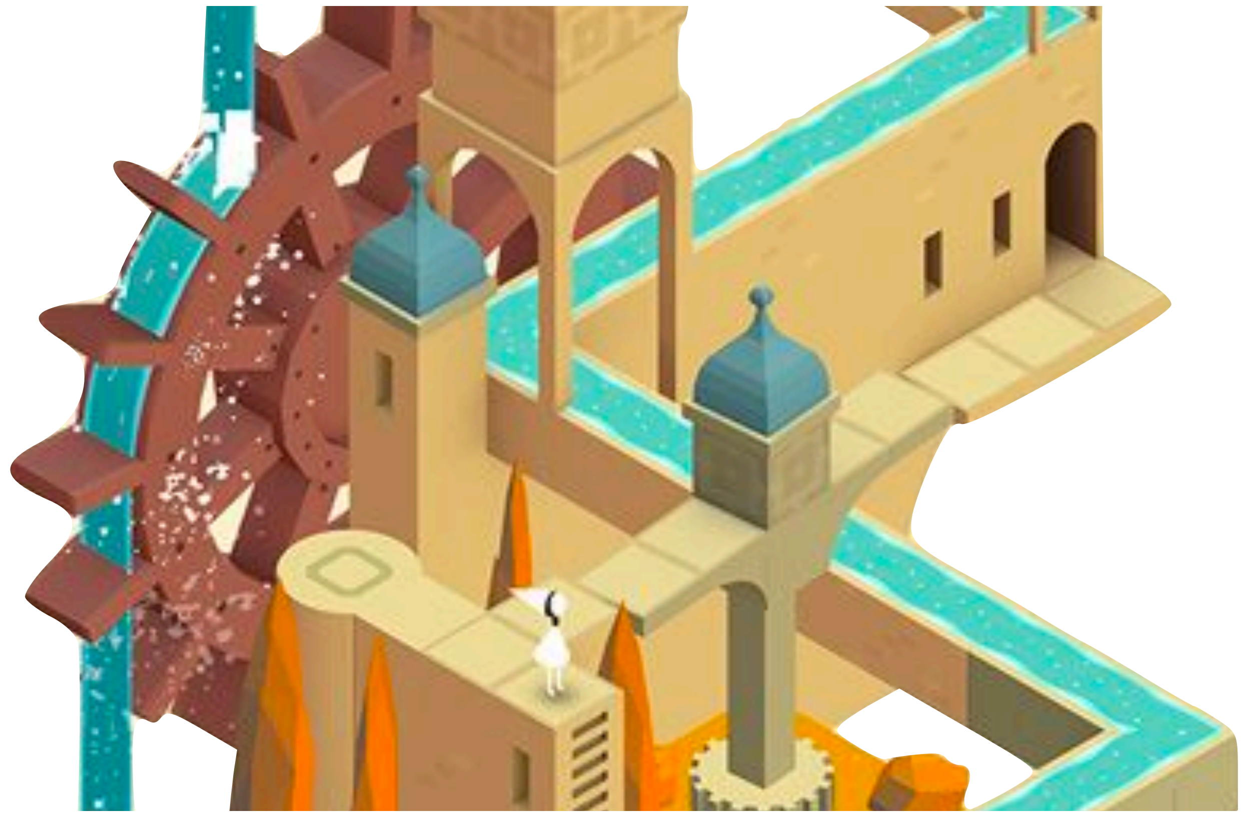
Joaquin Vanschoren (TU Eindhoven)
and the OpenML team

image credit: ustwo

Overview

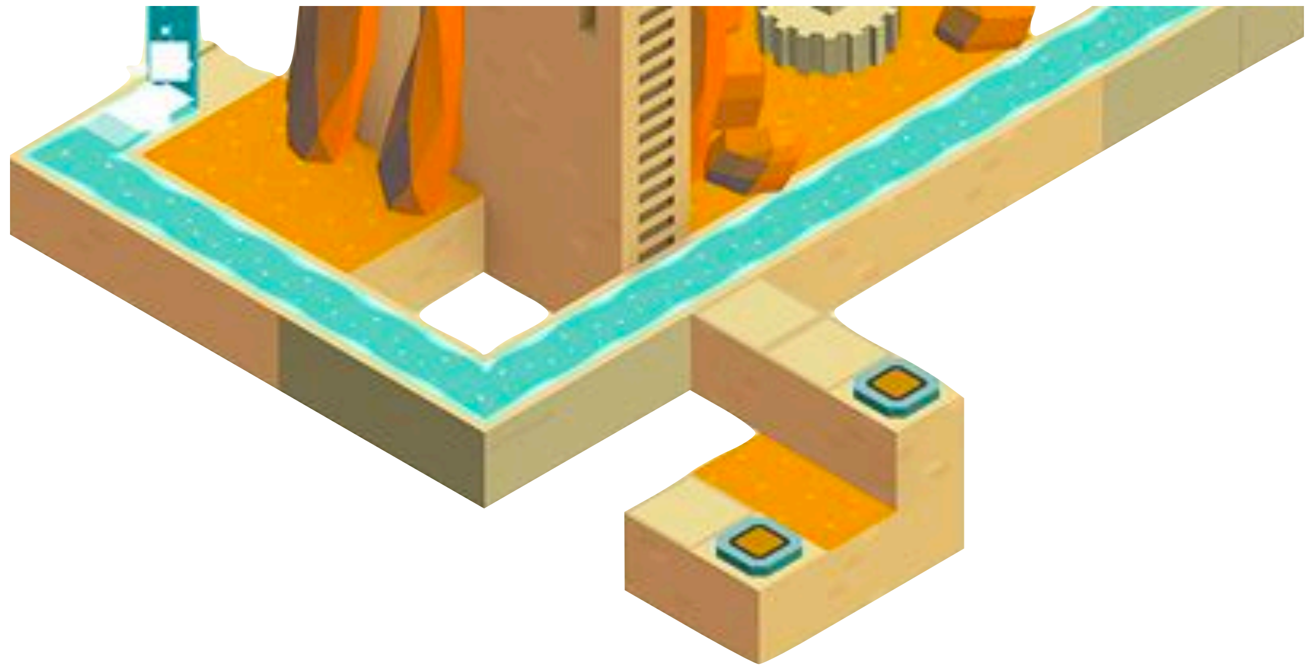
Part 1: *Why* democratize machine learning?

Easy access to ML (meta)data



Part 2: *How* self-learning AutoML (might) work

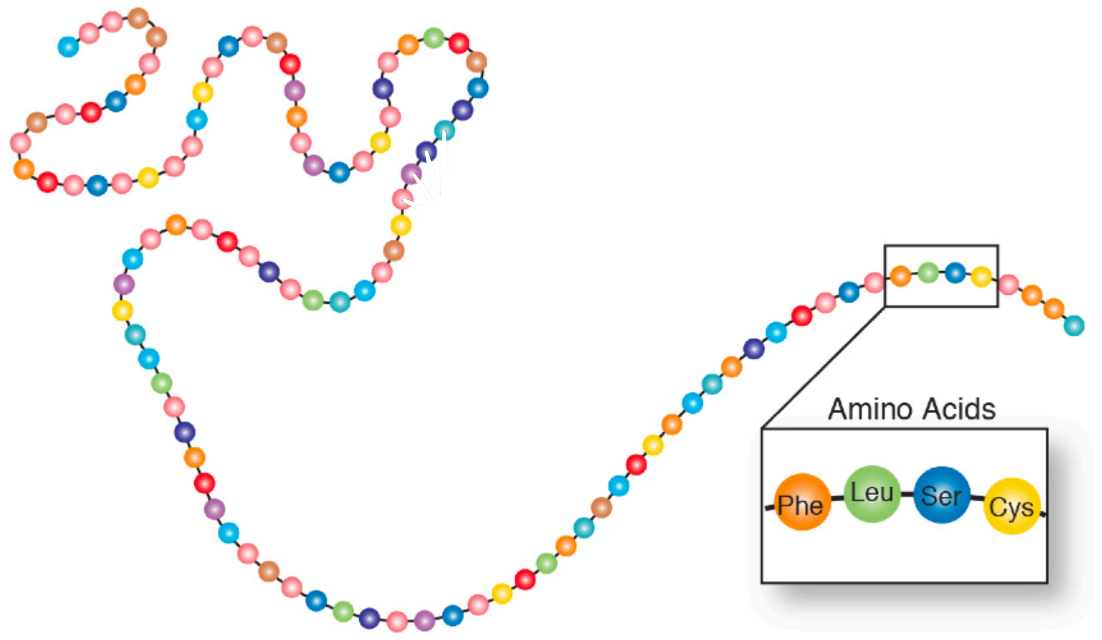
The machinery



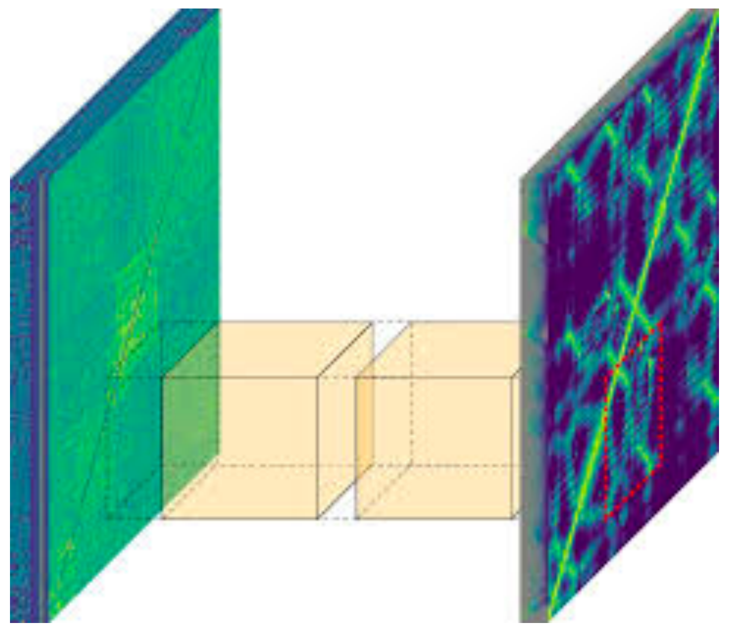
Part 3: The future and open challenges

Towards a virtuous cycle

AI can do amazing things

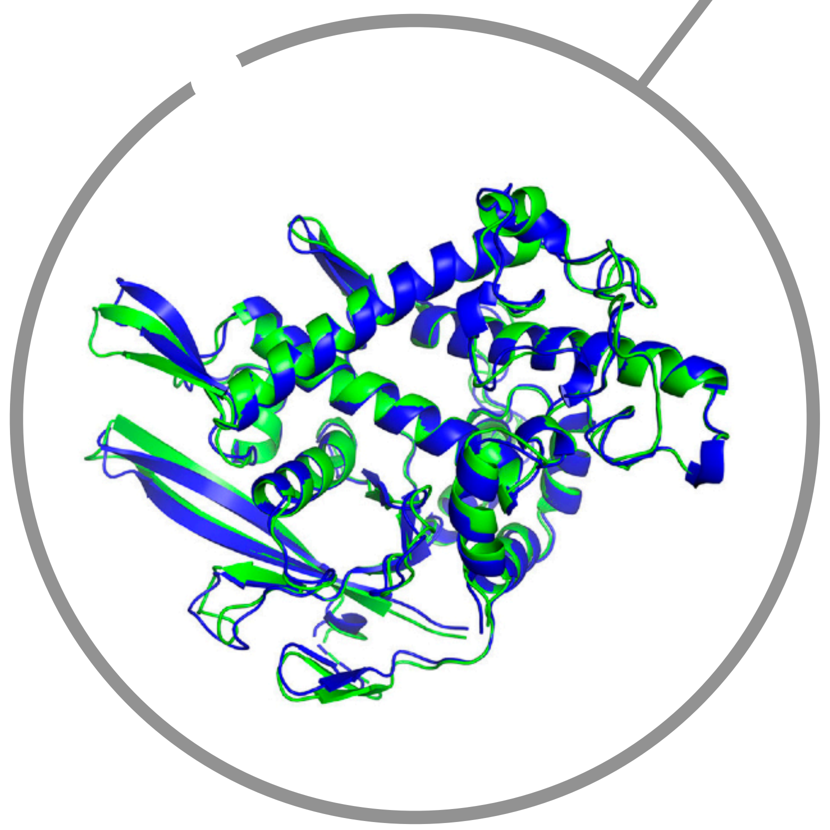


Scientific challenge
(protein sequence)



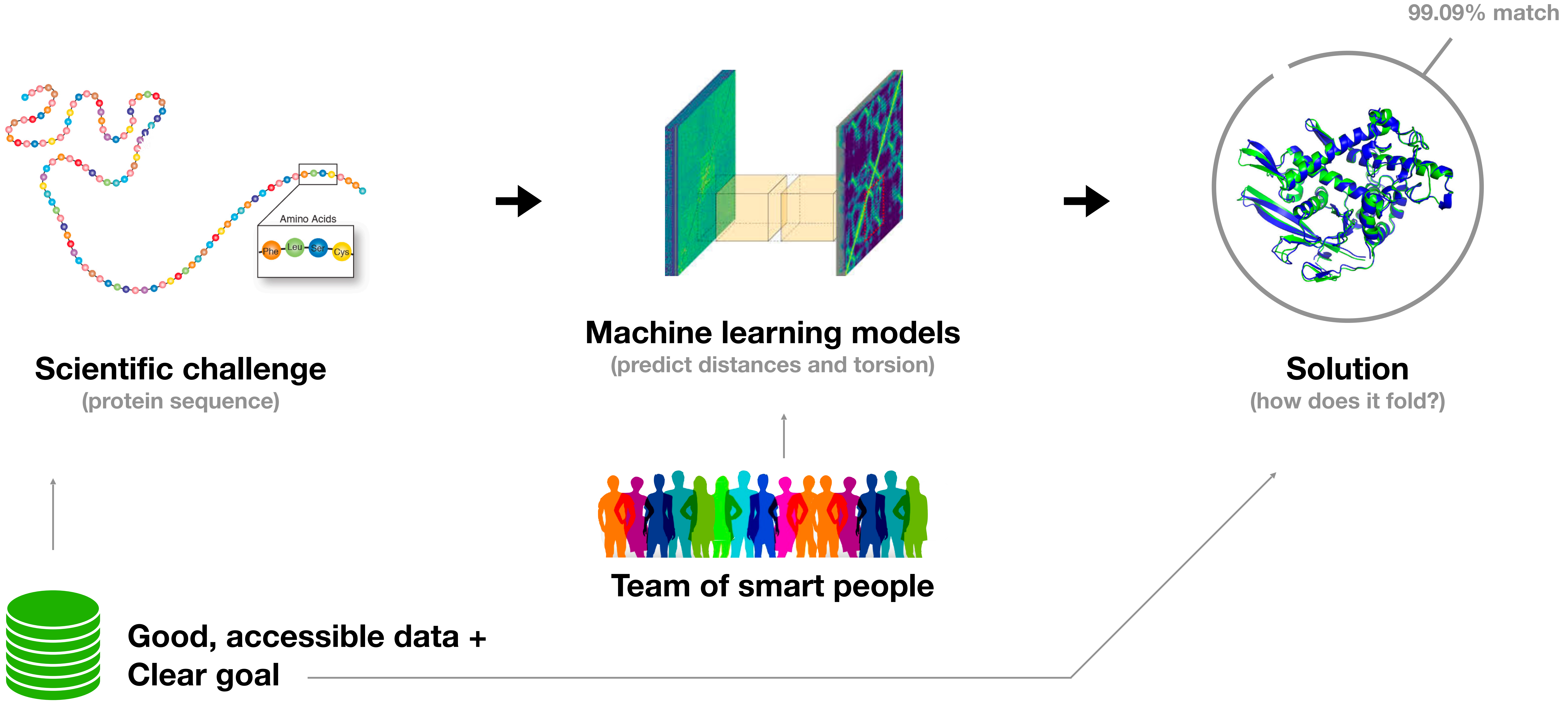
Machine learning models
(predict distances and torsion)

99.09% match



Solution
(how does it fold?)

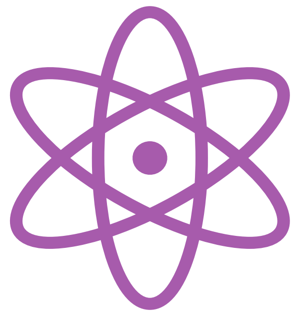
People + AI can do amazing things



Can we do this on a large scale?

Let's democratize data + AI

OK! How?



AI-ready data → **Team of smart people** → **Solution**

well-organized,
easily accessible,
uniformly formatted,
consistent meta-data

→

Smart people + automated AI tools

(*many* such teams)

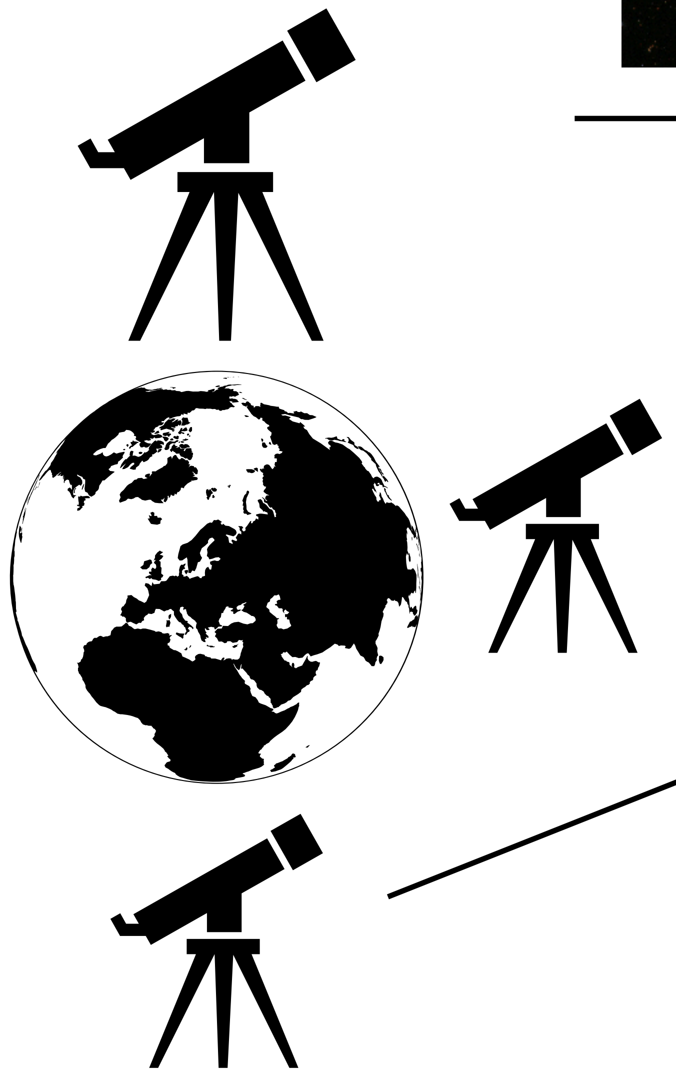
→

clear goal,
reproducible results

Democratizing data

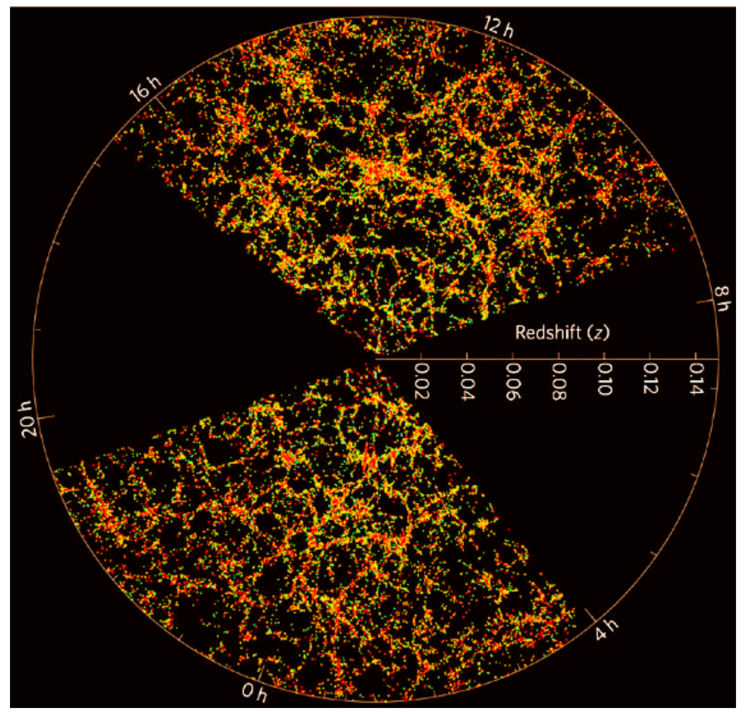
mapping the universe

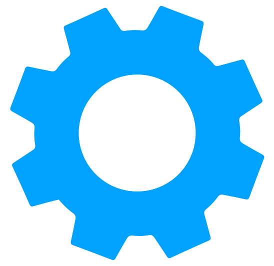
Democratizing data



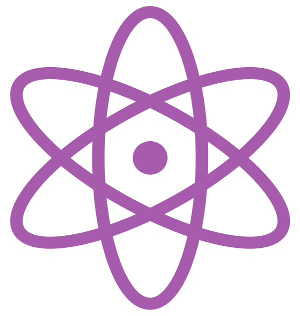
map of the universe

Well-organized data, easily accessible, uniform meta-data





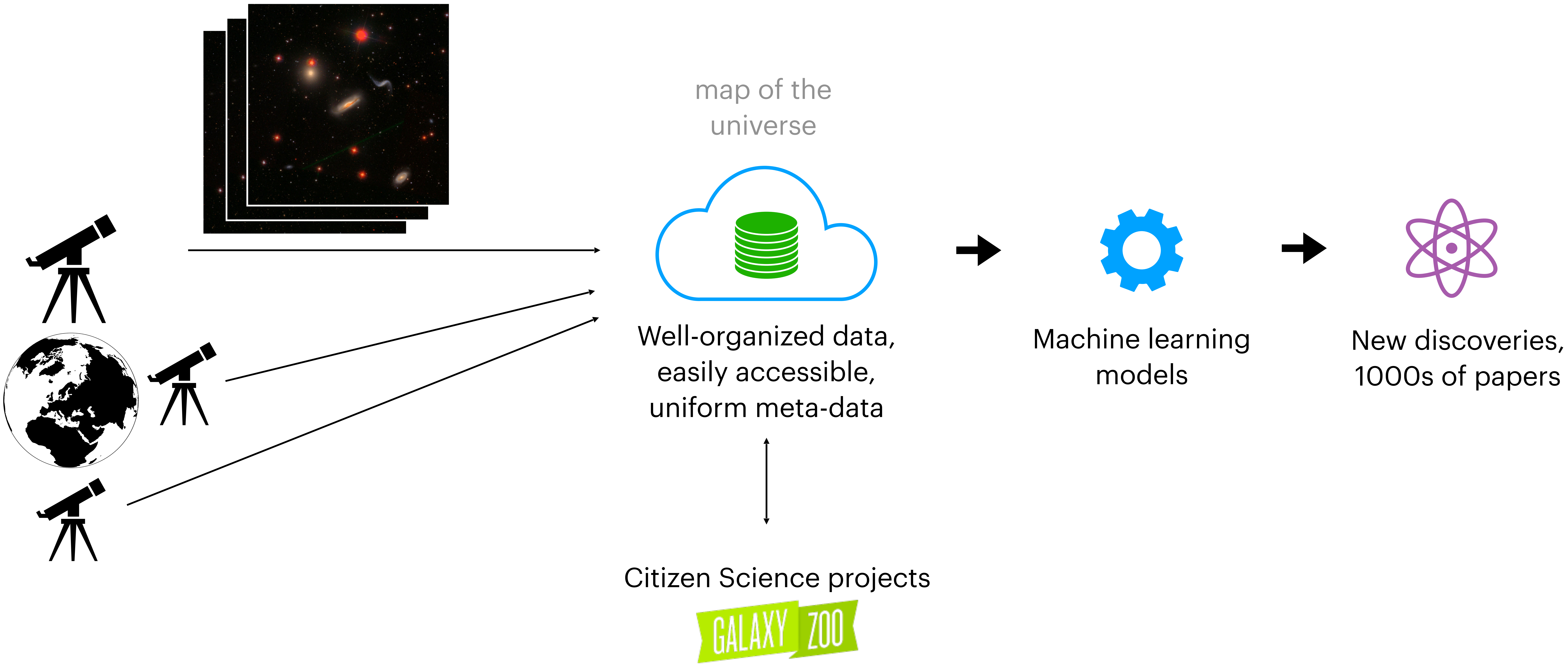
Machine learning models



New discoveries, 1000s of papers

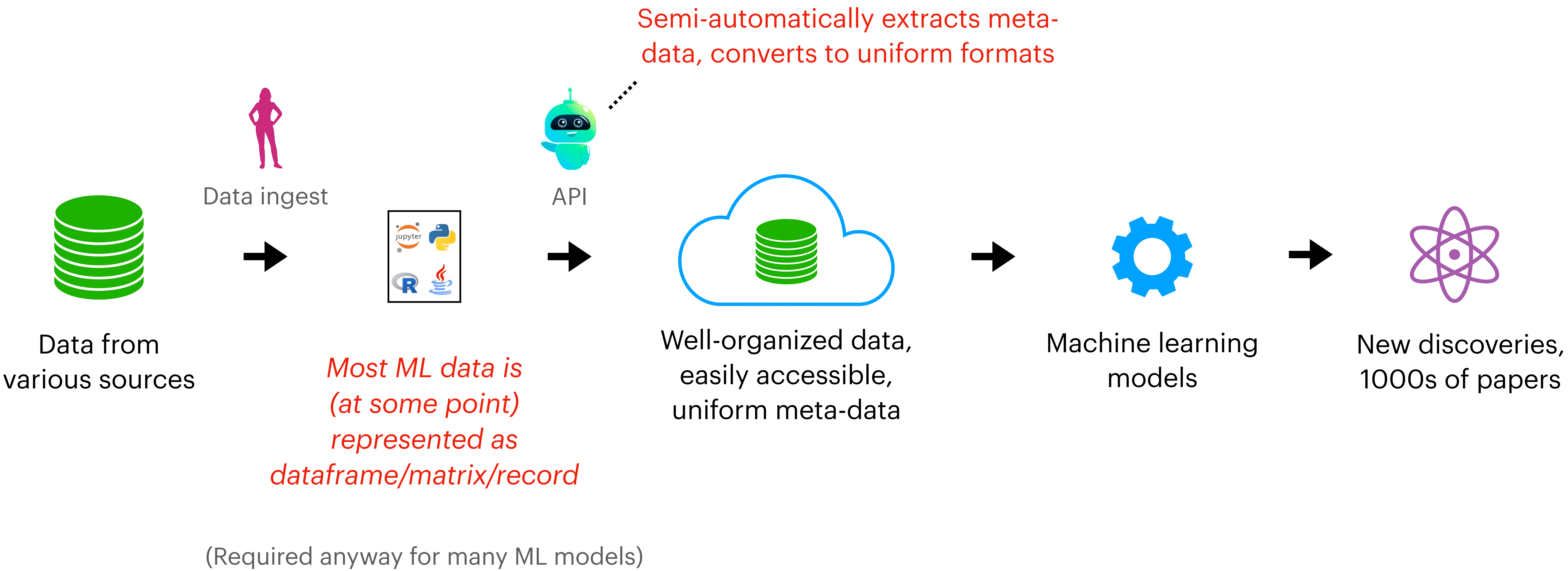
Democratizing data

How can we generalize this idea to data of any kind?

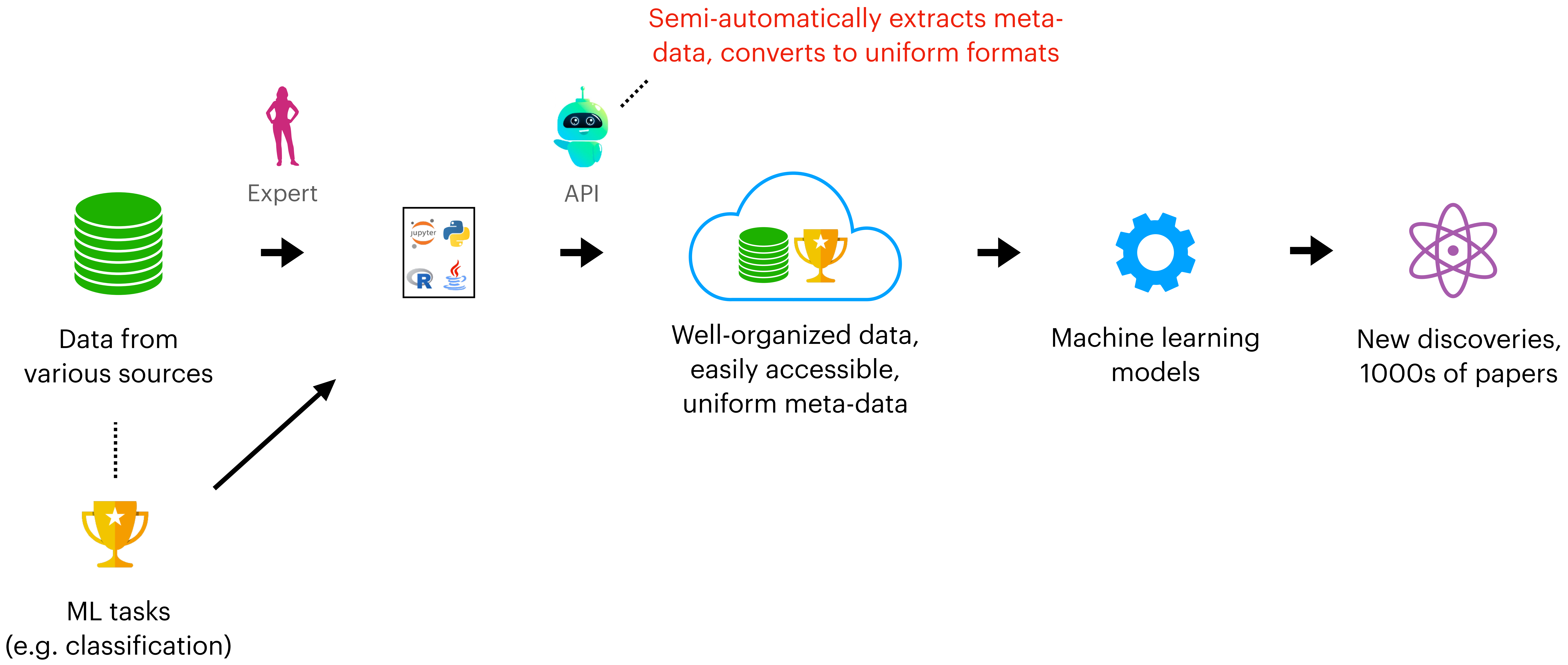


Democratizing ML data

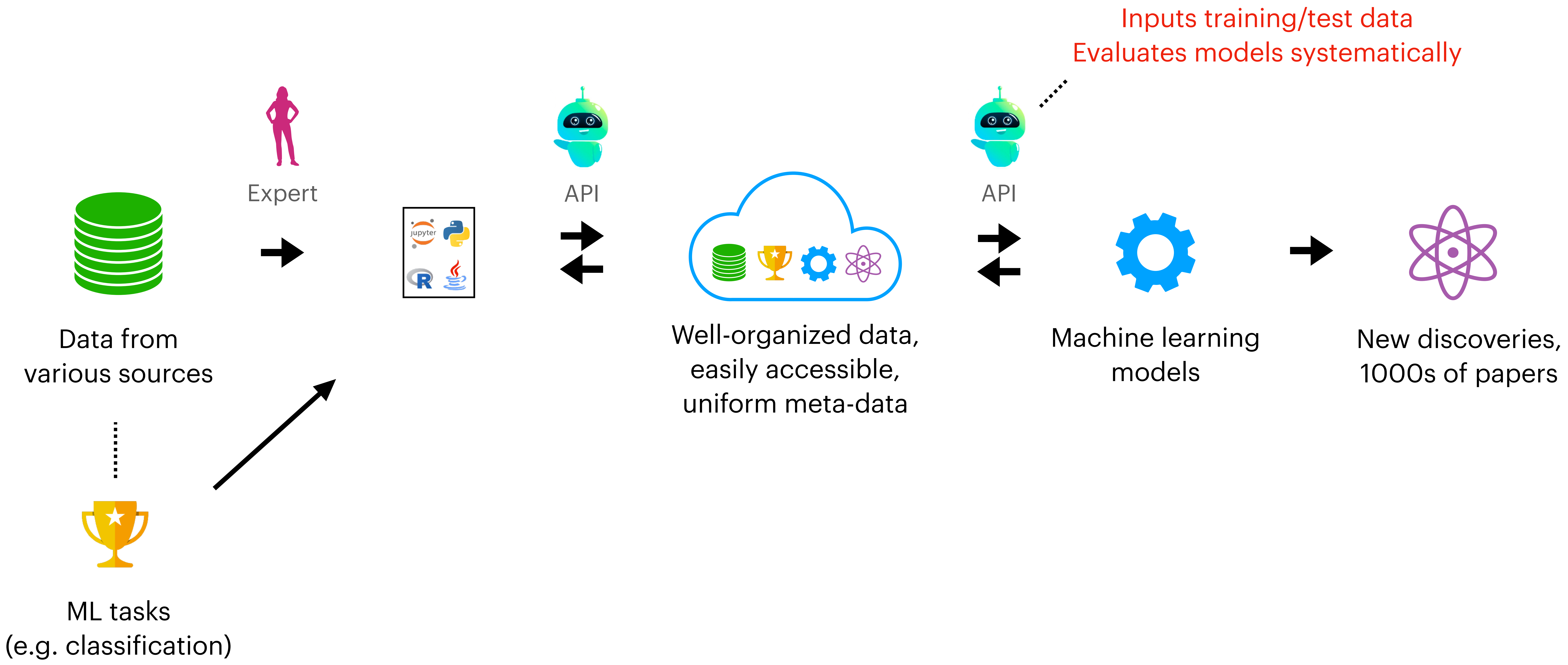
Democratizing ML data



Democratizing ML data



Democratizing ML data

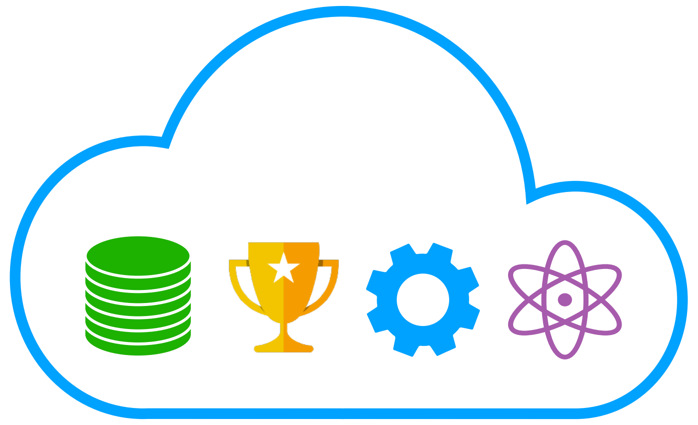


OpenML

An open platform for discovering and sharing ML datasets, algorithms, experiments

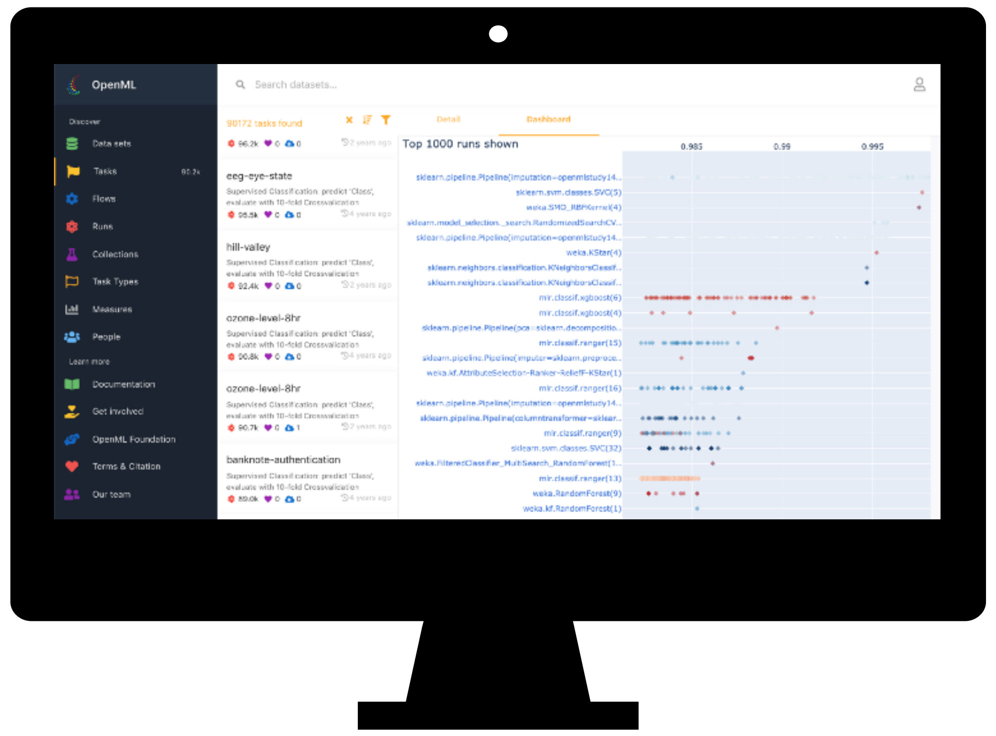
Accessible from anywhere, anytime
(scripts, notebooks, apps, cloud jobs)

API



OpenML

API



Website
(new.openml.org)

OpenML web interface

Search

Datasets

Dataset analysis

OpenML web interface

Tasks

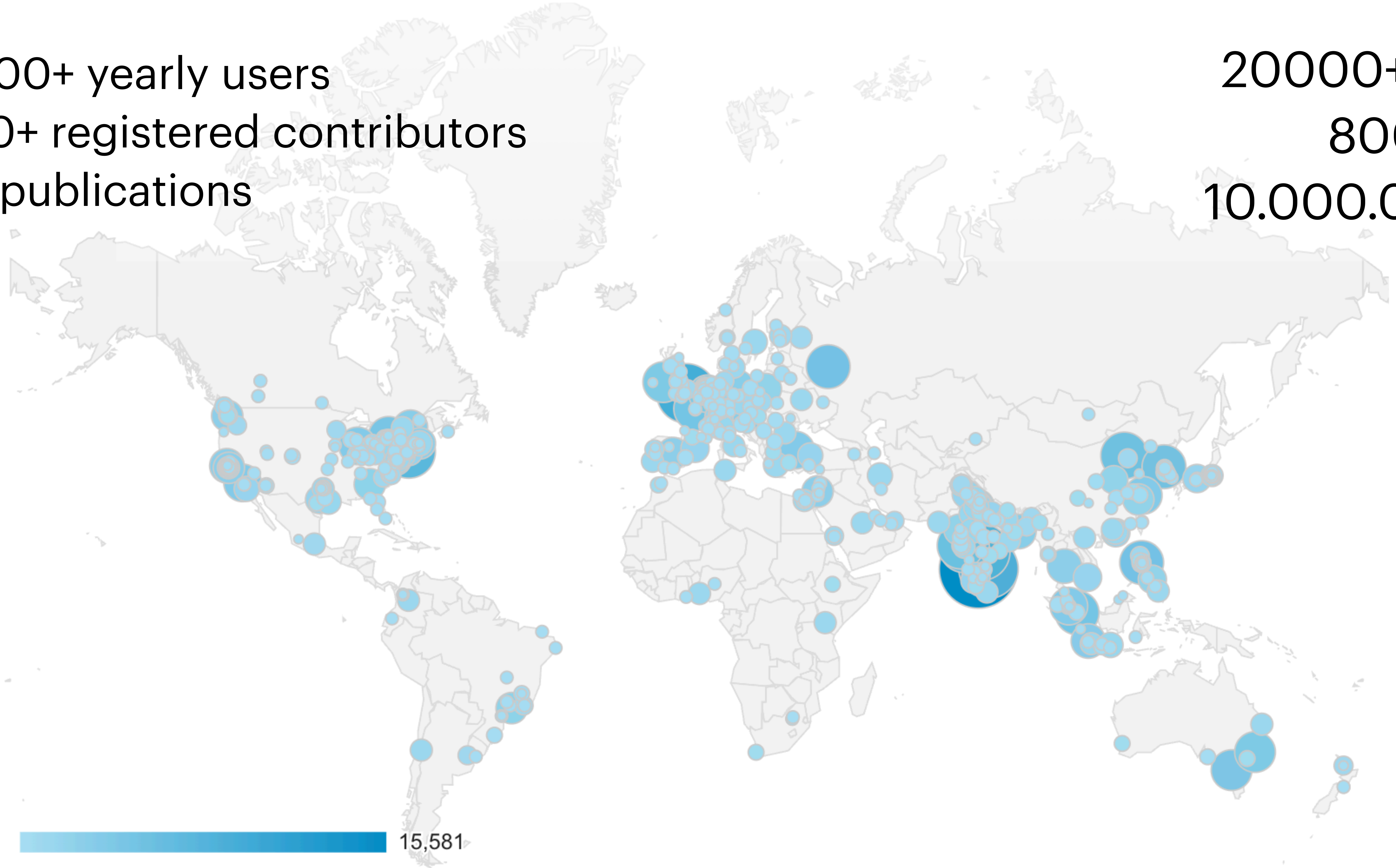
Algorithms

Evaluations (every dot is a model)

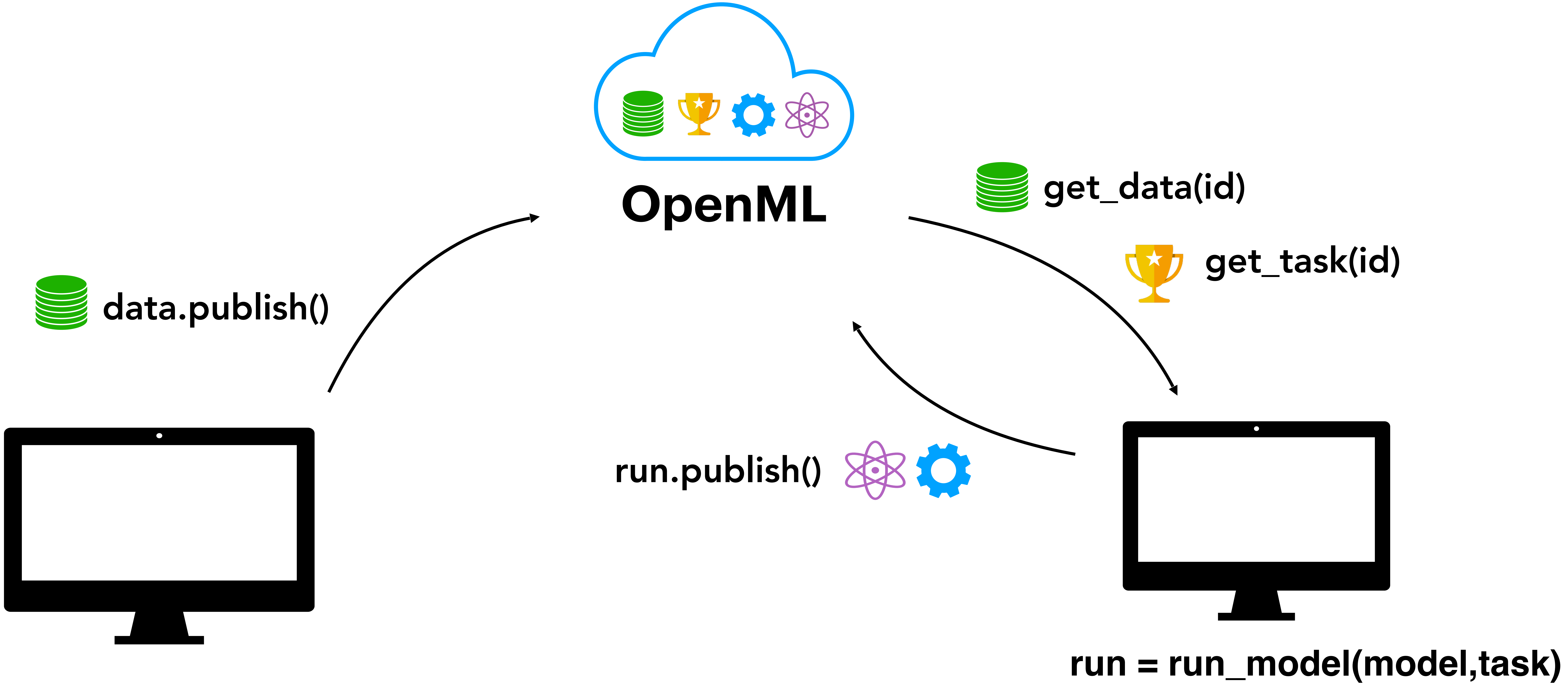
OpenML Community

250000+ yearly users
13000+ registered contributors
700+ publications

20000+ datasets
8000+ flows
10.000.000+ runs



Frictionless machine learning



APIs:

Integrations:

Frictionless machine learning

```
from sklearn import ensemble
from openml import tasks, runs

model = ensemble.RandomForestClassifier()
task = tasks.get_task(3954)
run = runs.run_model_on_task(model, task)
run.publish()
```

integrations

Frictionless machine learning

```
import torch.nn
from openml import tasks, runs

model = torch.nn.Sequential(
  processing_net, features_net, results_net)
task = tasks.get_task(3954)
run = runs.run_model_on_task(model, task)
run.publish()
```

integrations

Frictionless machine learning

Datasets
(+ rich meta-data)

size, dimensions
type, provenance
versions,…

Tasks
(problem meta-data)

what to learn,
how to evaluate

Flows
(algorithm meta-data)

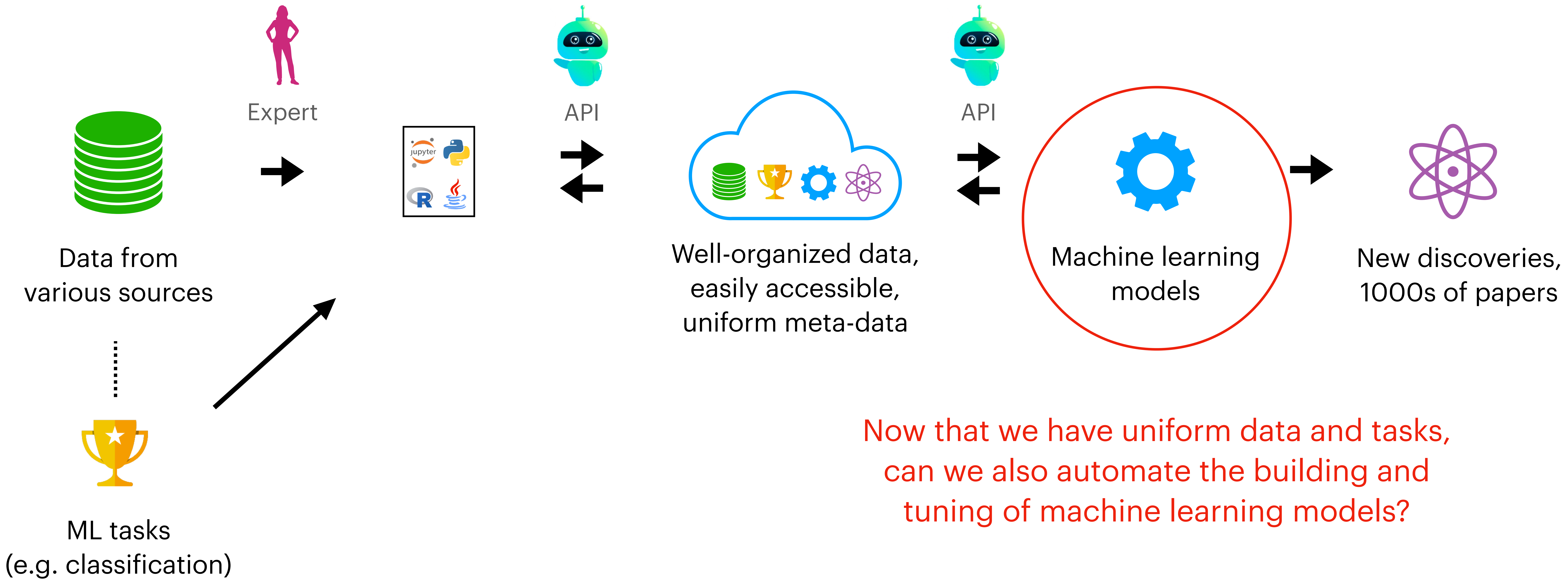
dependencies,
detailed structure
(pipelines, neural nets)

Runs
(model meta-data)

configurations,
evaluations

collected automatically, to ensure reproducibility

Democratizing machine learning itself

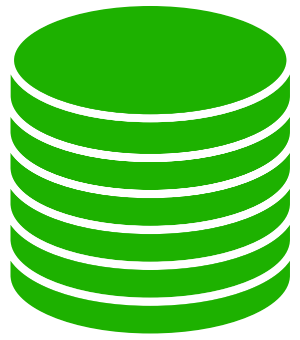


Now that we have uniform data and tasks,
can we also automate the building and
tuning of machine learning models?

Why is Machine Learning labor-intensive?

Machine learning pipelines / models have an **infinite** range of possibilities (many still unknown)

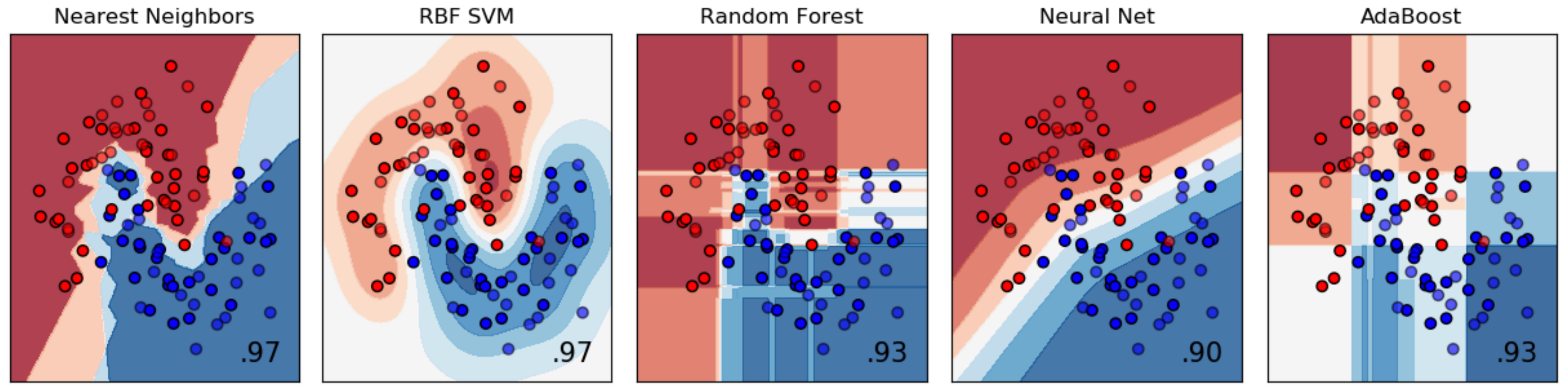
Requires implicit knowledge



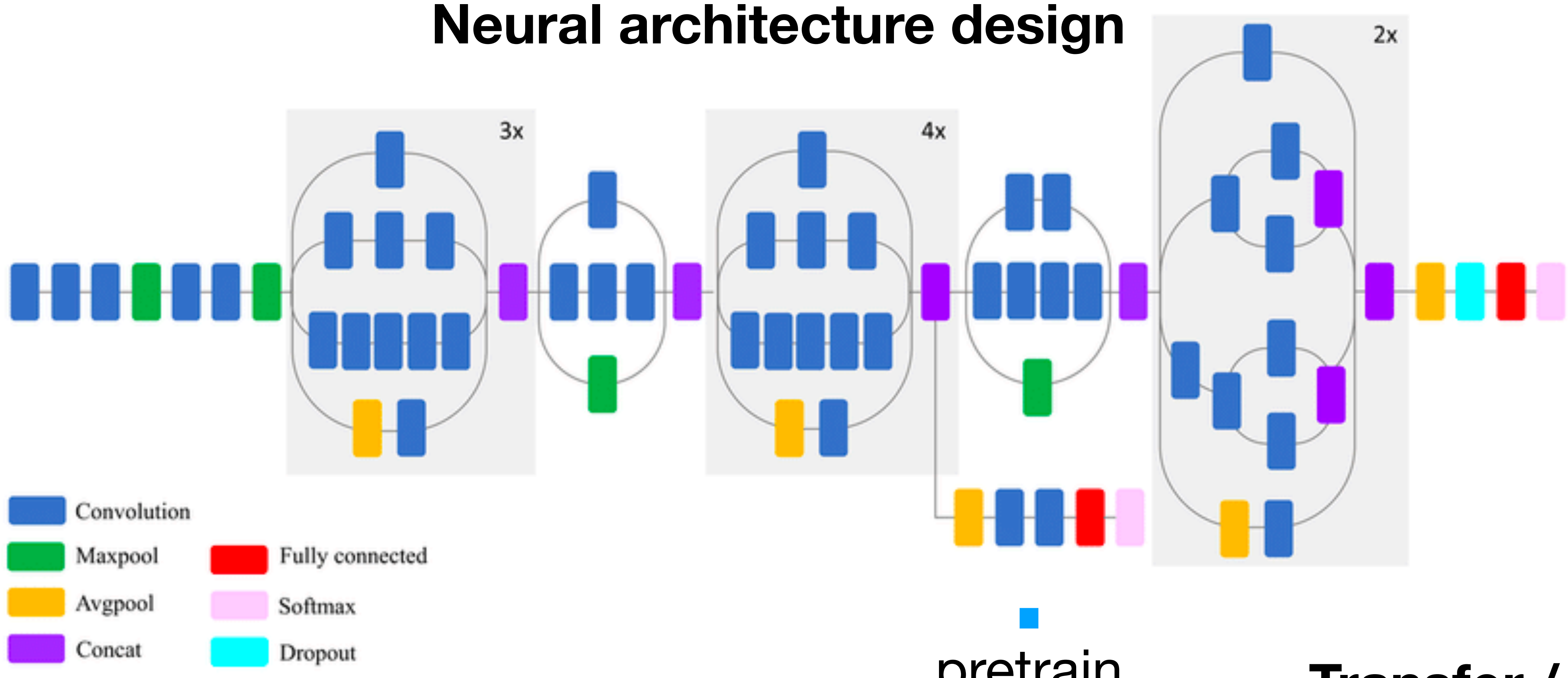
Data

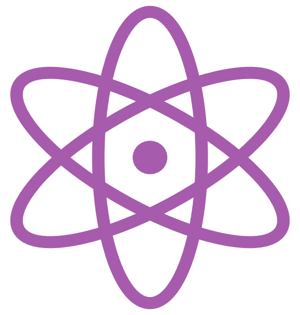
cleaning, preprocessing, featurization, selection,…

Model selection



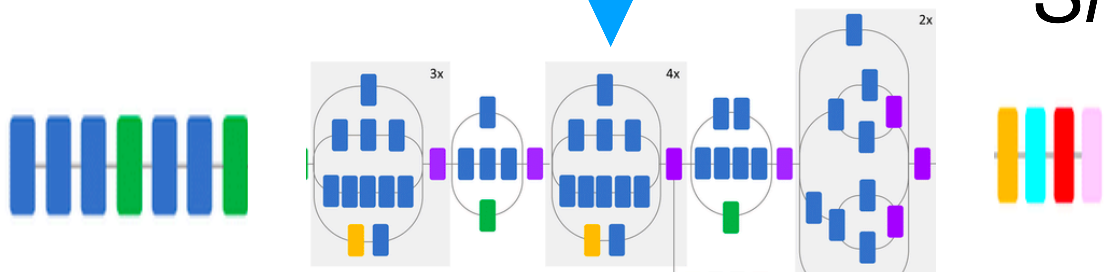
Neural architecture design





Solution

pretrain



Transfer / continual learning

Small data, few-shot learning

Can we automate this process and share implicit knowledge?

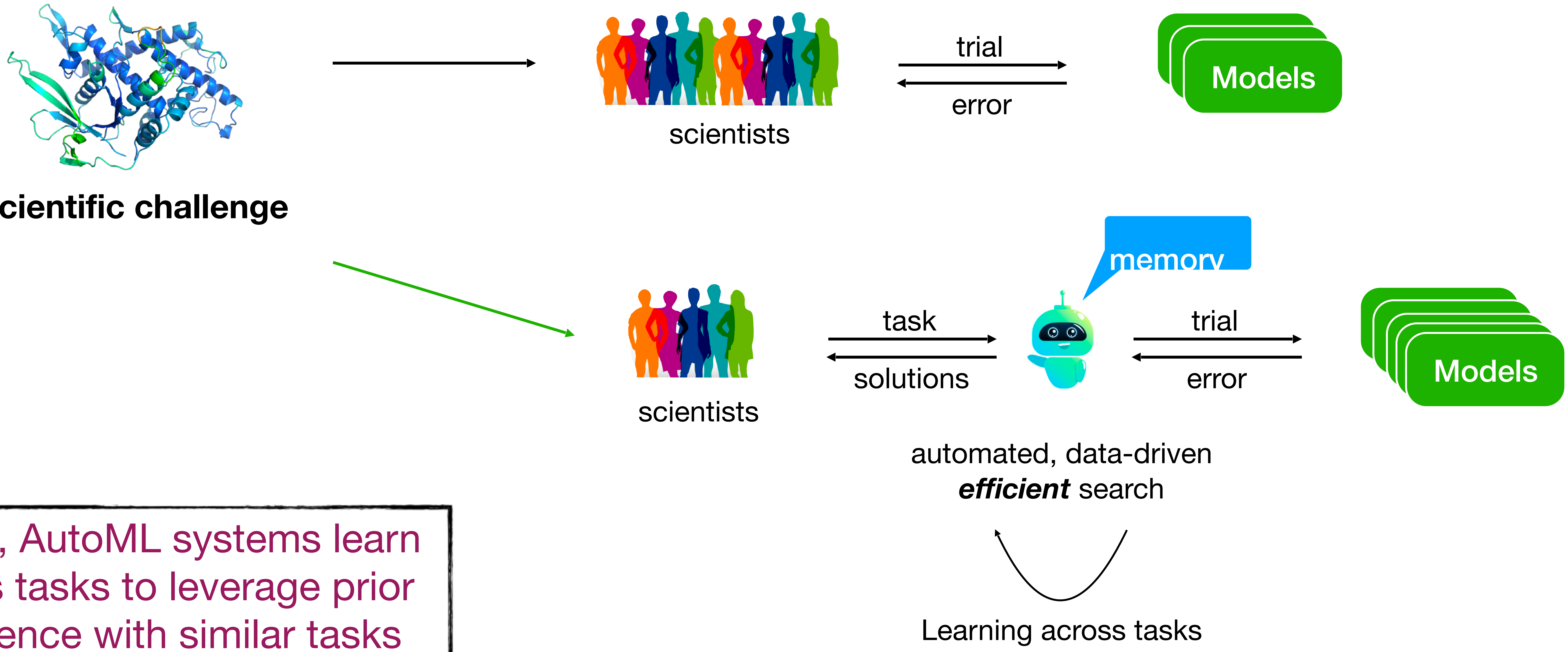
Also needed for robust autonomous systems

> Antenna broken. No communication with Earth.

Sigh… I’ll have to learn this by myself

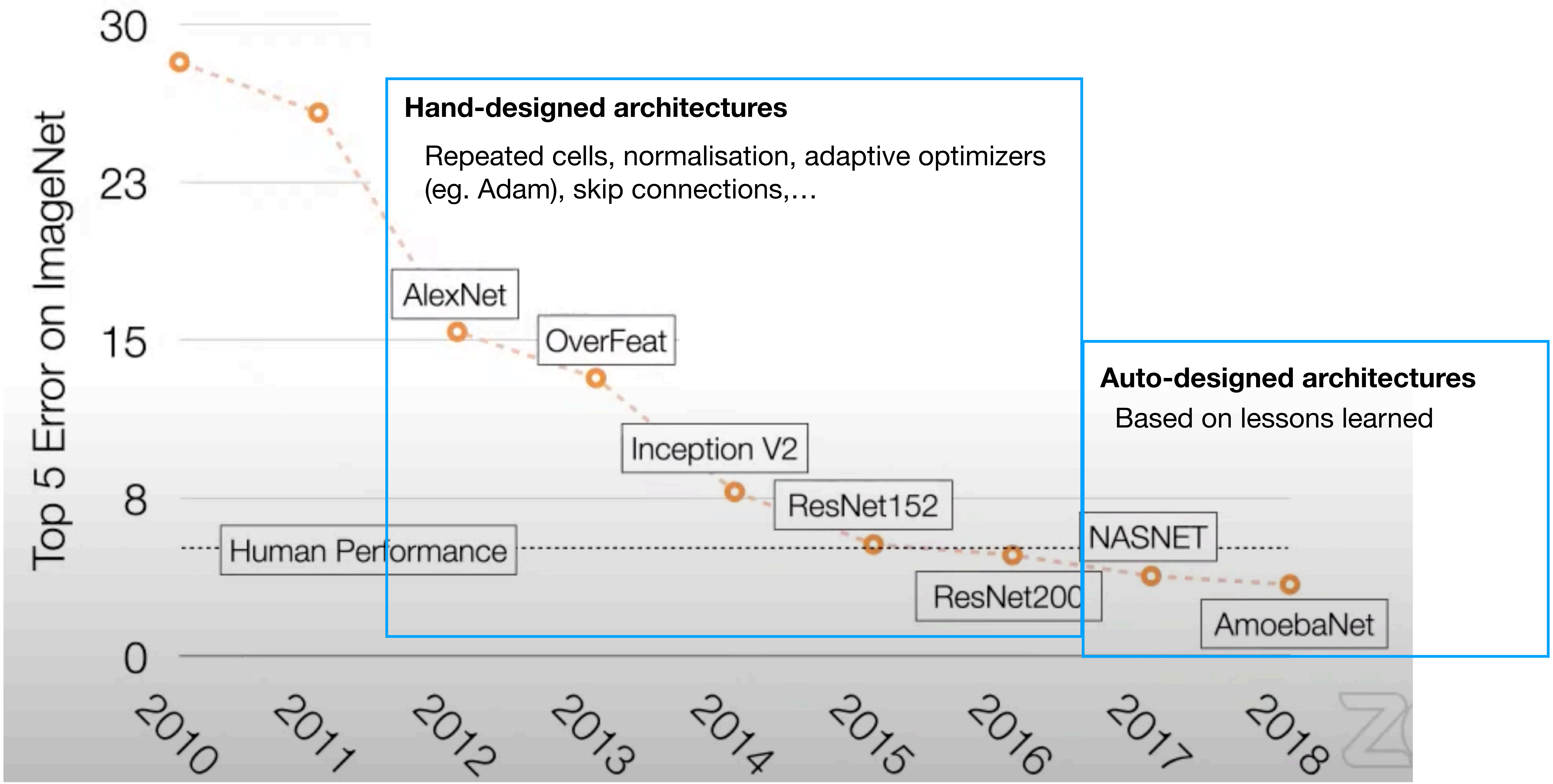
Automatic Machine Learning (AutoML)

Replace manual trial and error with automated search (based on prior experience)



Ideally, AutoML systems learn across tasks to leverage prior experience with similar tasks

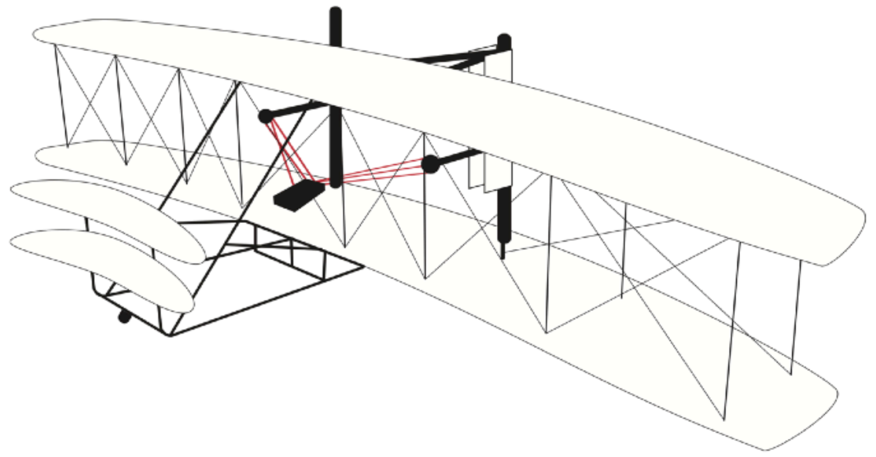
What drove progress?



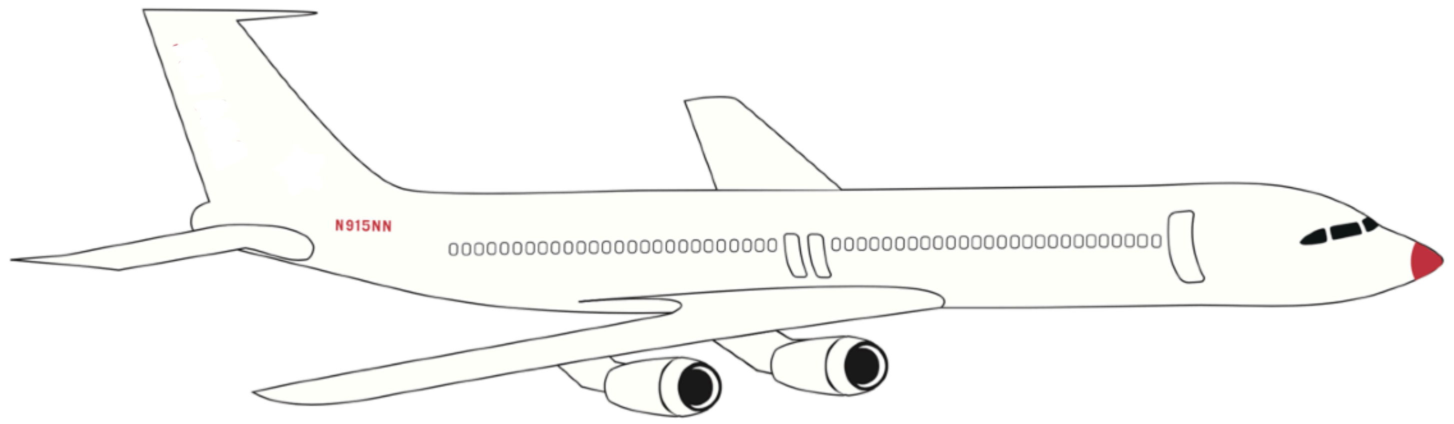
A (key) part of the wider AI challenge

Talwalkar, 2018

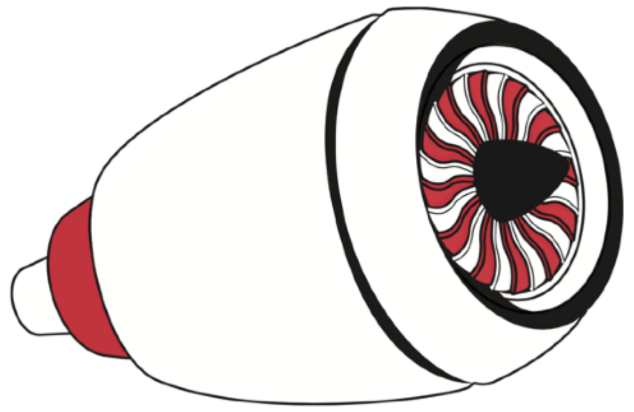
We're only in the pioneering age



1903: first powered controlled flight



1910: cultural acceptance, only for elites



1952: democratized flight

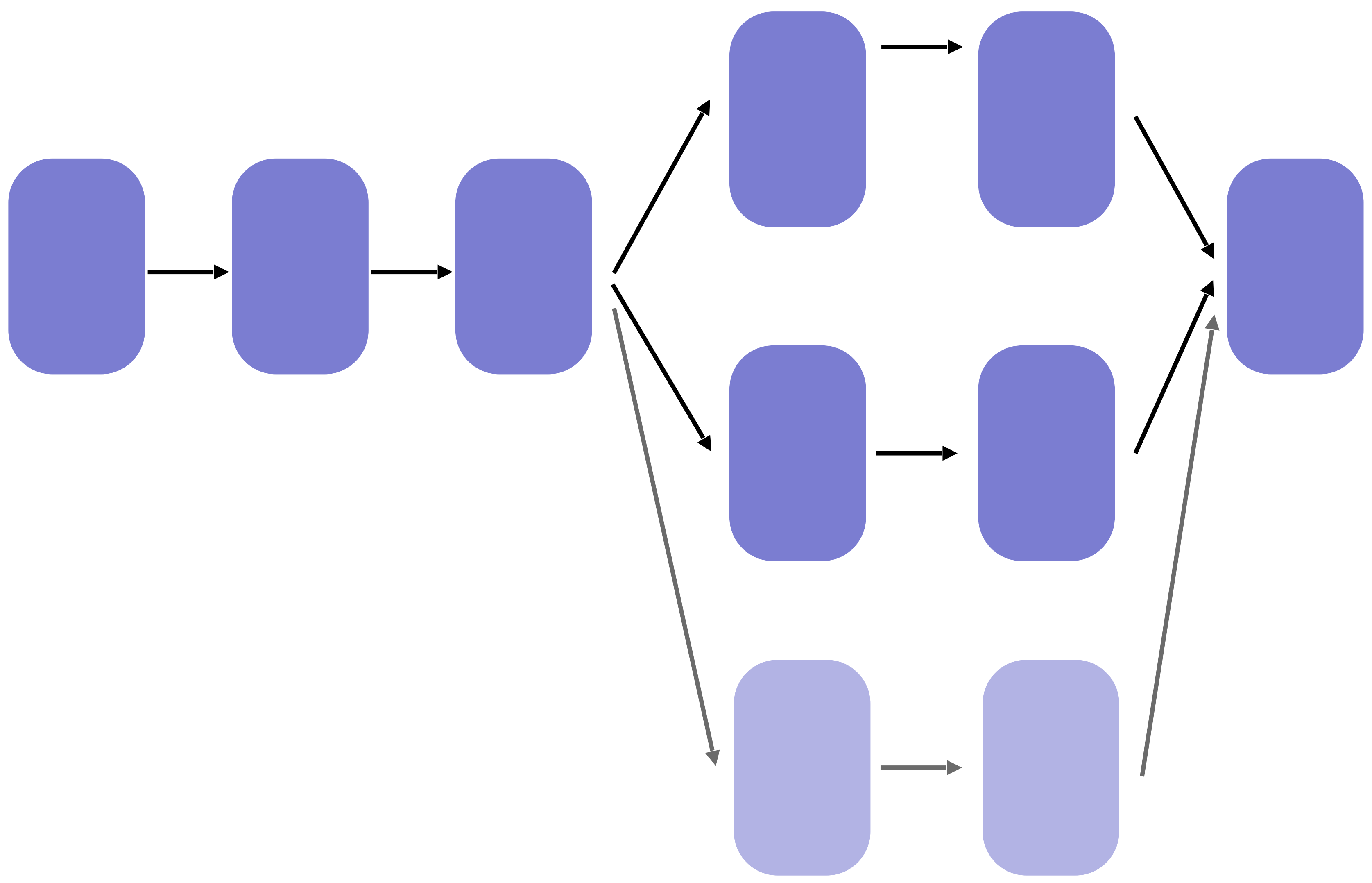
Efficiency, automation, safety
(Science + engineering)

Many challenges remain to democratize AI

- Efficiency: efficient algorithms (transfer, continual), hardware
- Automation: efficient, adaptive AutoML
- Safety: explainability, fairness, causal analysis

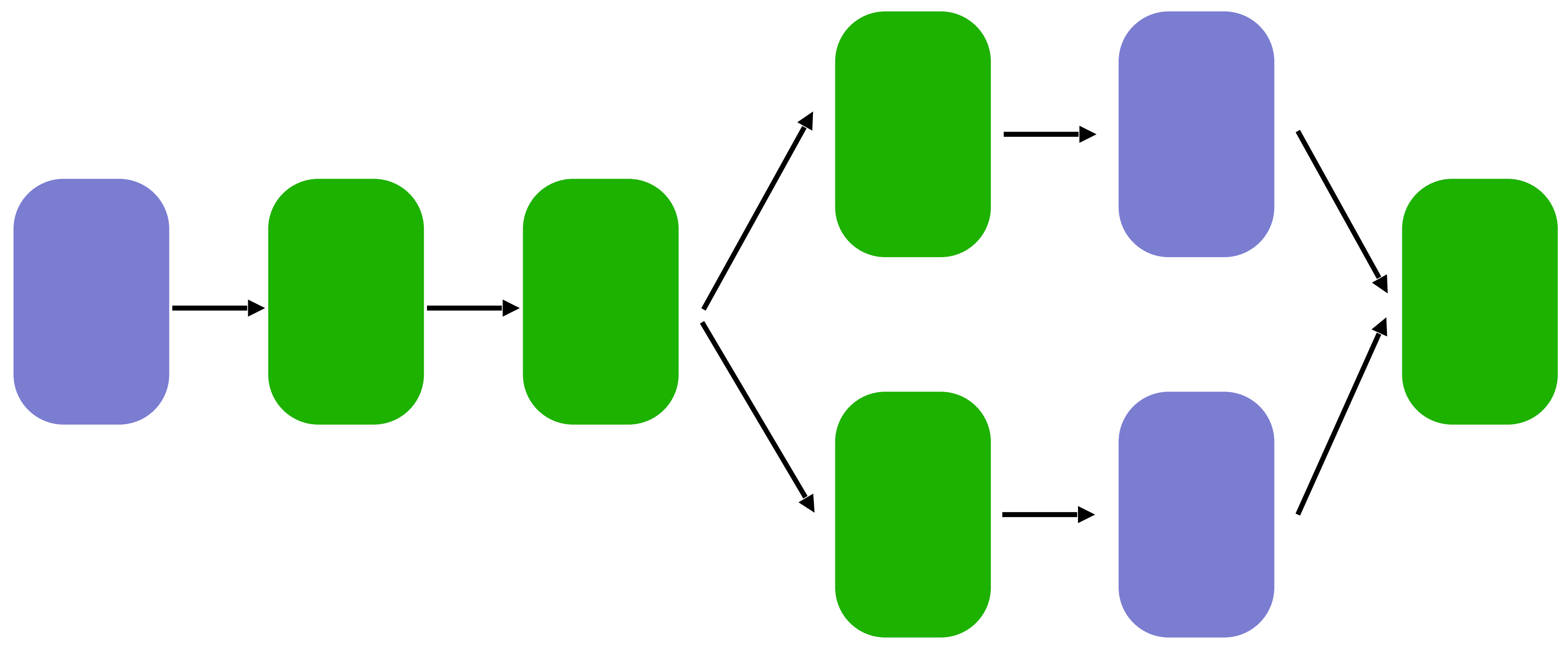
AutoML: subproblems

- **Architecture search space**: *represent* all pipelines or neural architectures
  - Pipeline operators, neural layers, interconnections,…
  - Defines a (complex) search space



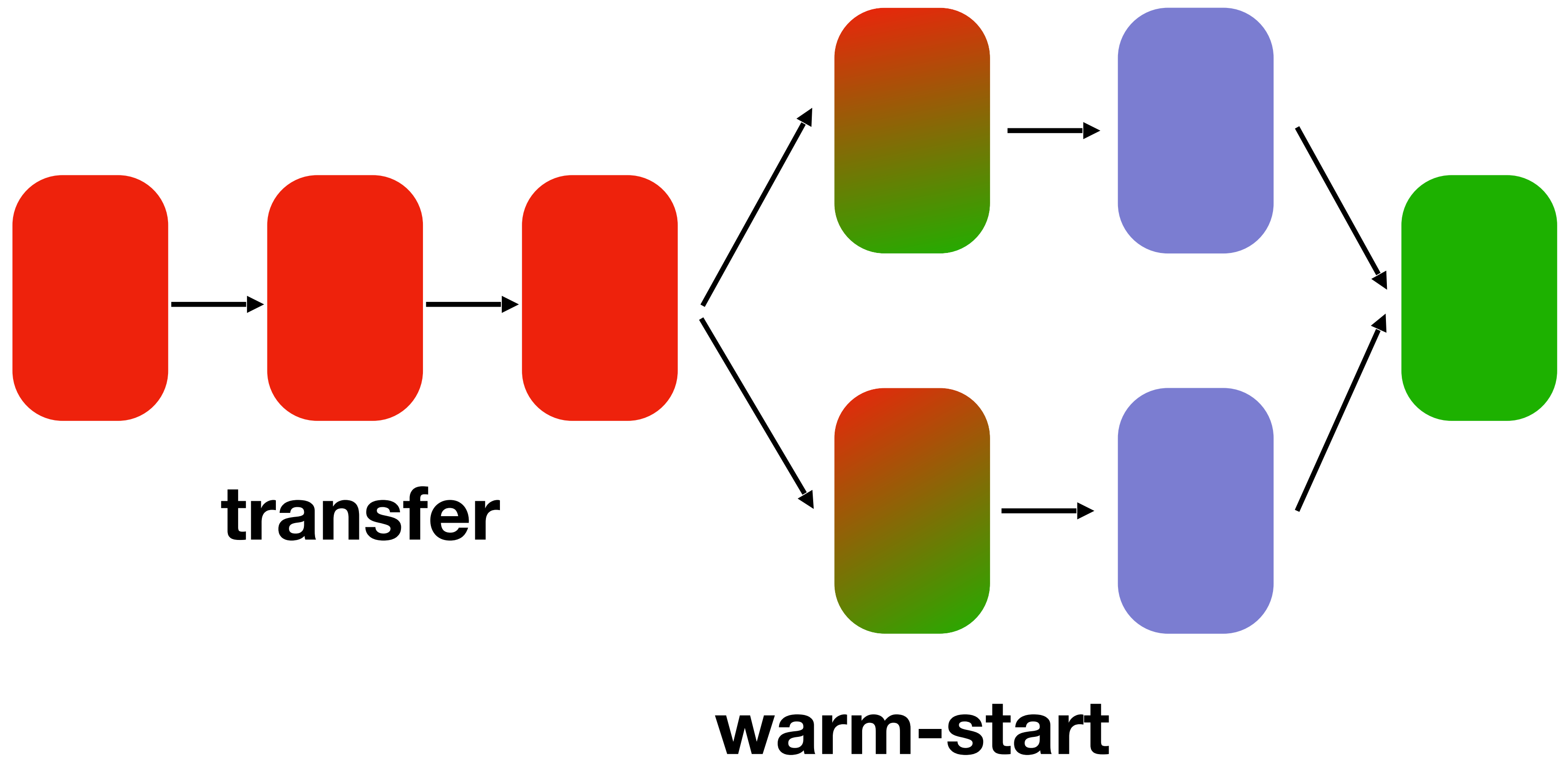
AutoML: subproblems

- **Architecture search space**: *represent* all possible architectures
- **Optimization**:
  - What is the best architecture? Which options are important? How to optimise?
  - Which method? Evolution, Gradient-based, Bayesian Optimization, Reinforcement learning, …



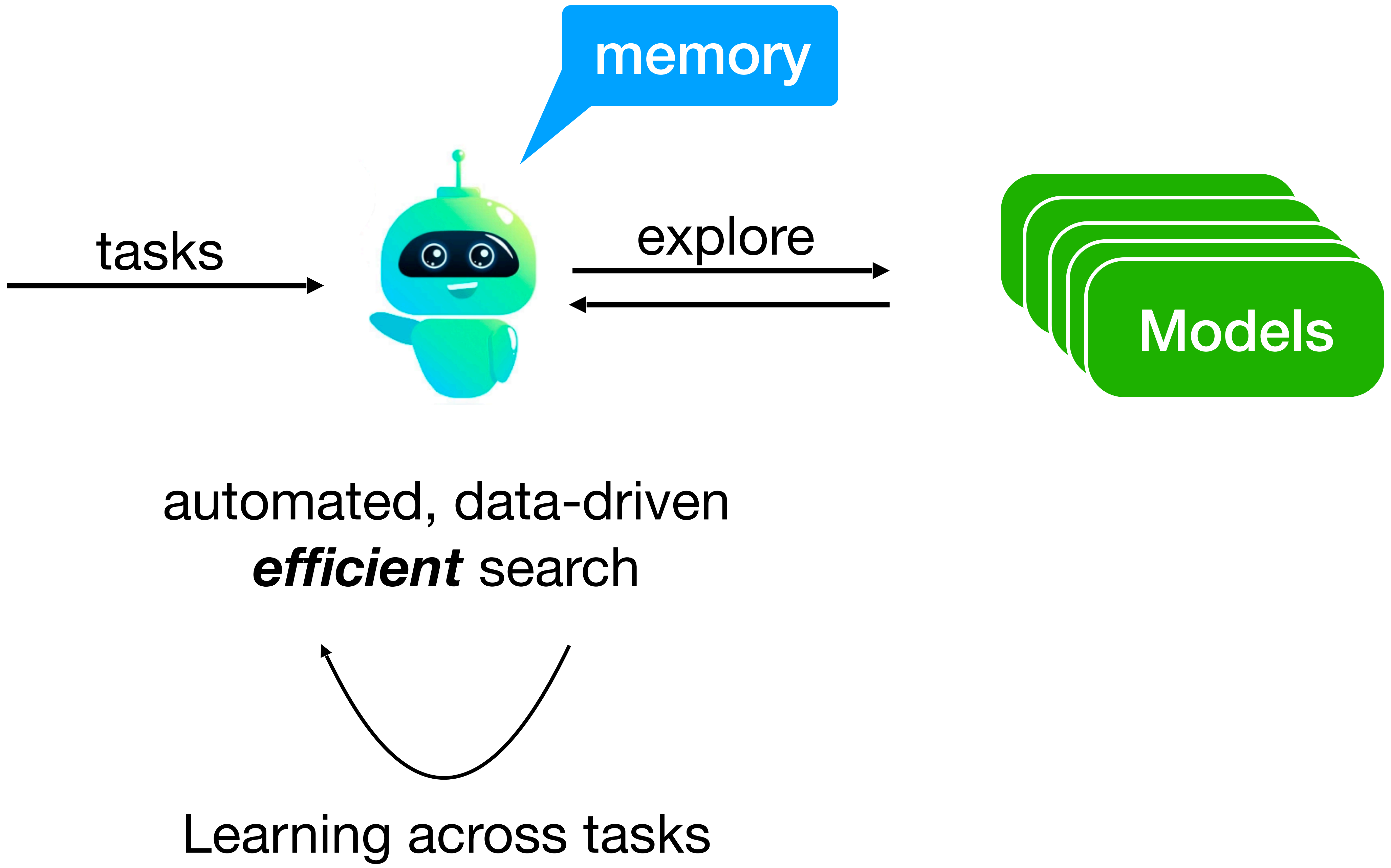
AutoML: subproblems

- **Architecture search space**: *represent* all possible architectures
- **Optimization**: *optimize* architecture and hyperparameters
- **Meta-learning**: how can we transfer *experience* from previous tasks?
  - Don't start from scratch (search space is too large)
  - Transfer learning: reuse good architectures/configurations/weights
  - Warm starting: start from promising architectures/configurations/initializations



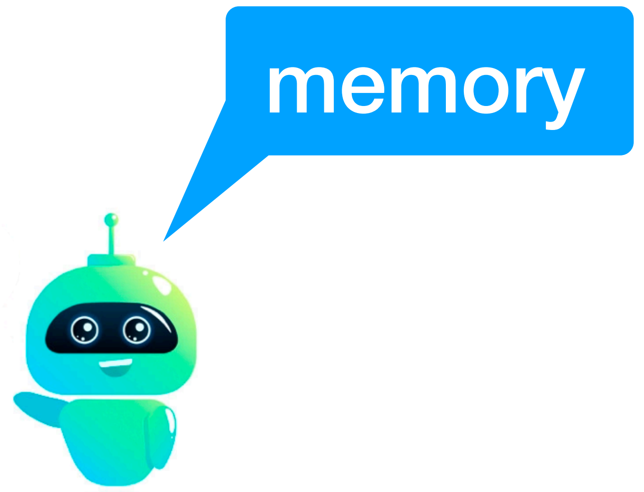
Learning AutoML systems

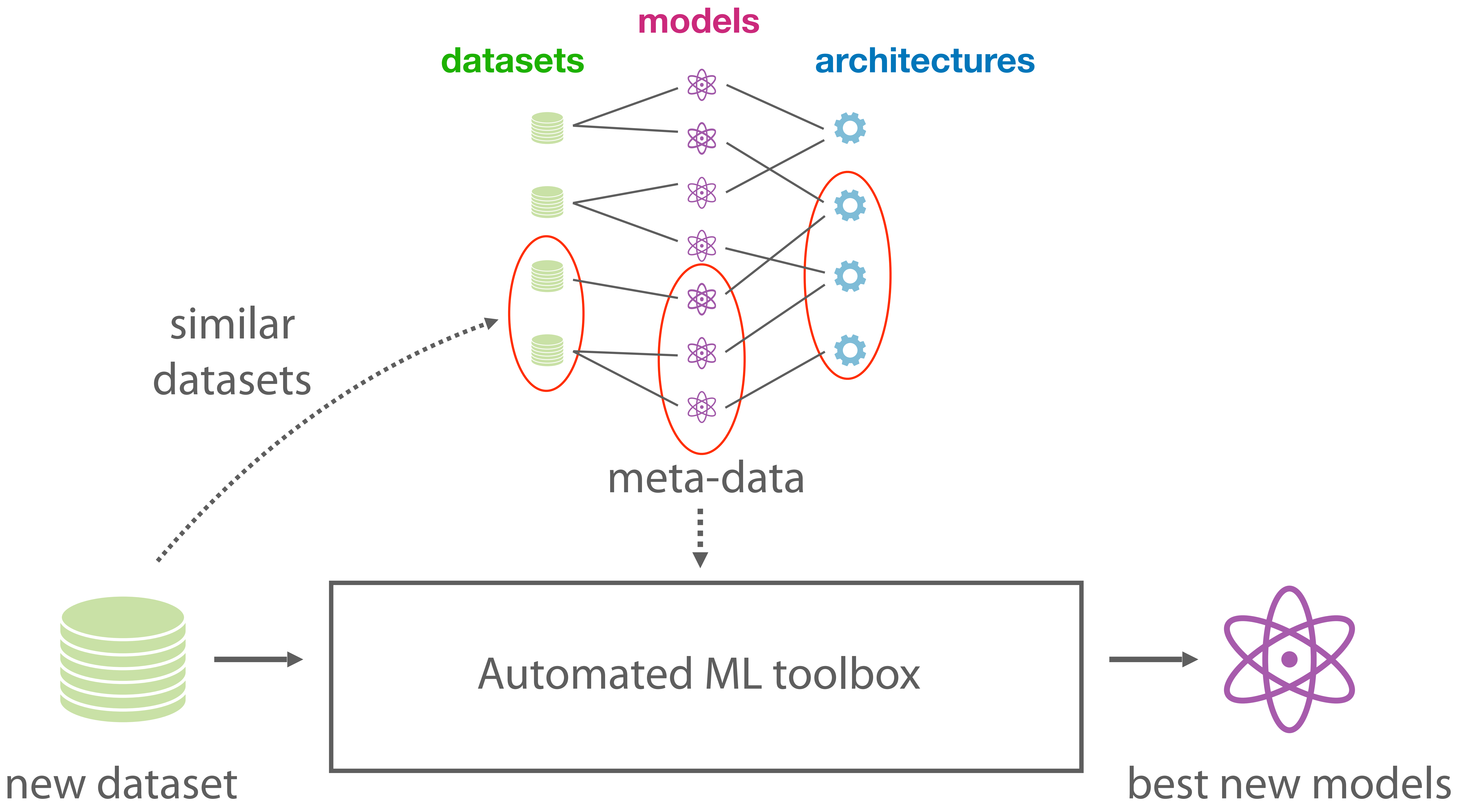
Start a virtuous cycle by letting AutoML systems learn across tasks to leverage prior experience



OpenML as a global memory

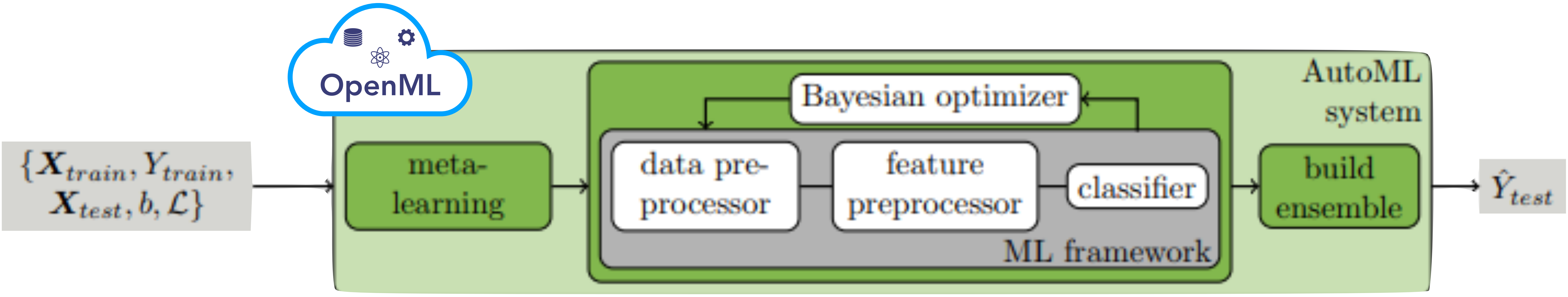
Machine-readable repository of machine learning results





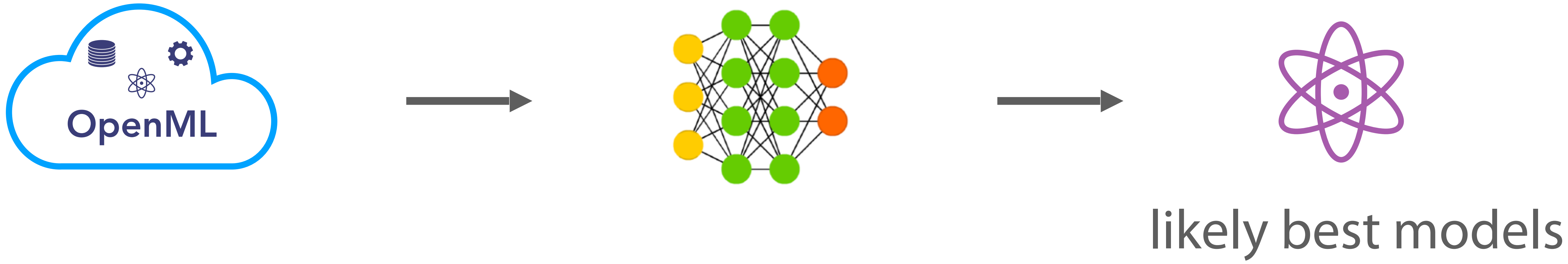
Automating machine learning

auto-sklearn: uses OpenML to *warm-start* the search for the best pipelines



Feurer et al. 2020

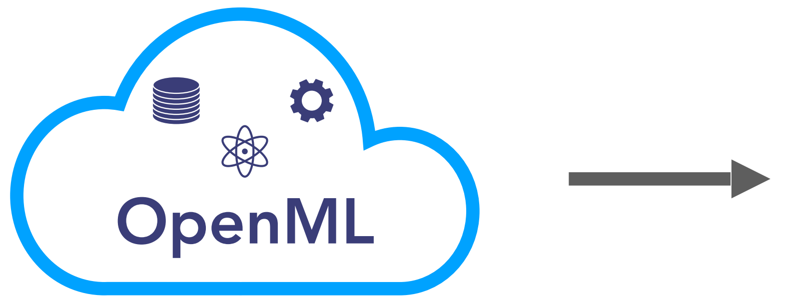
ABLR (Amazon): uses OpenML to learn how to search hyperparameters

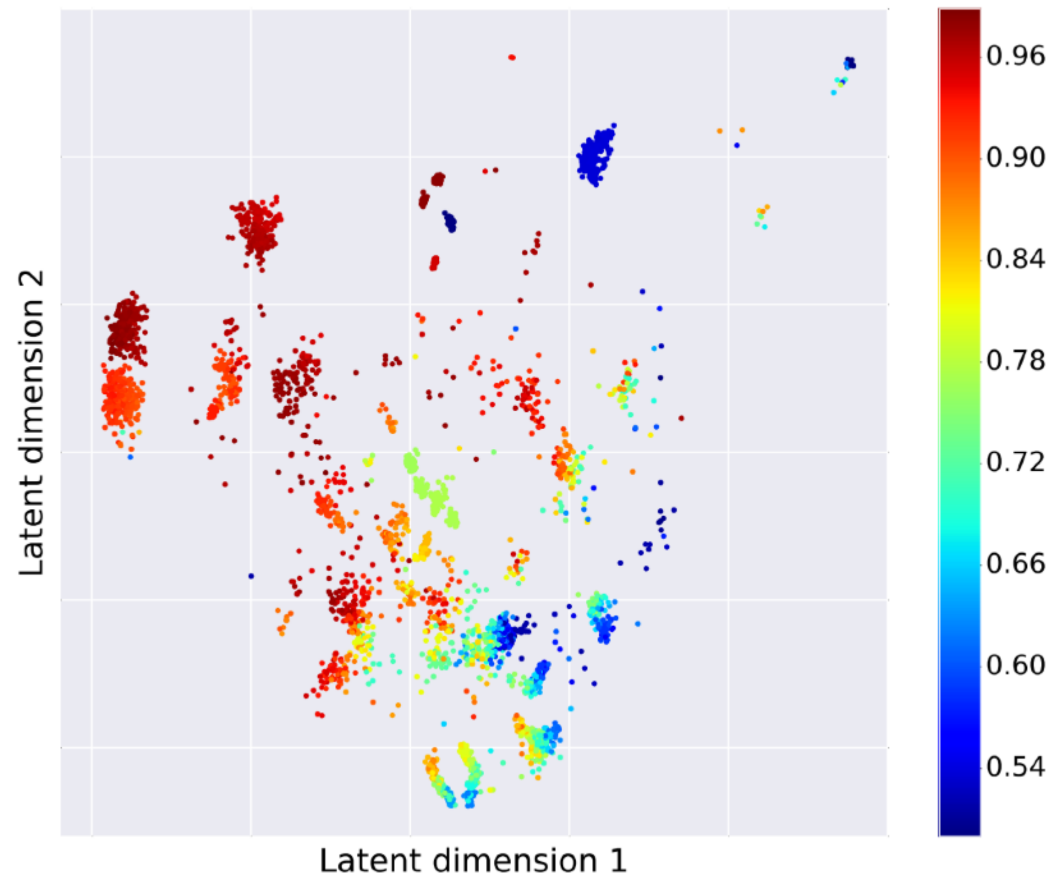


Perrrone et al. 2018

Automating machine learning

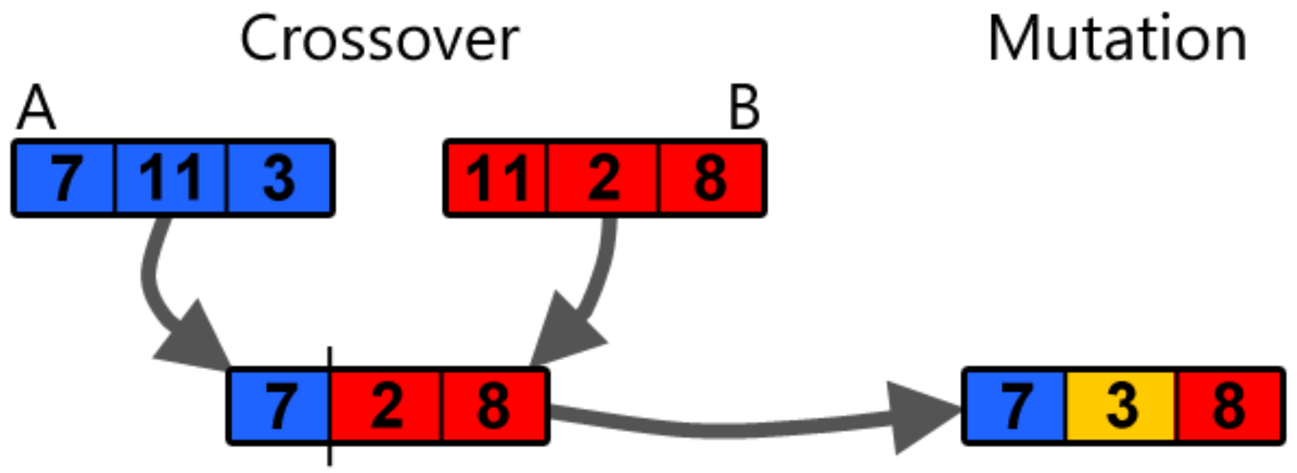
ProbMF (Microsoft): uses OpenML to recommend the best algorithms

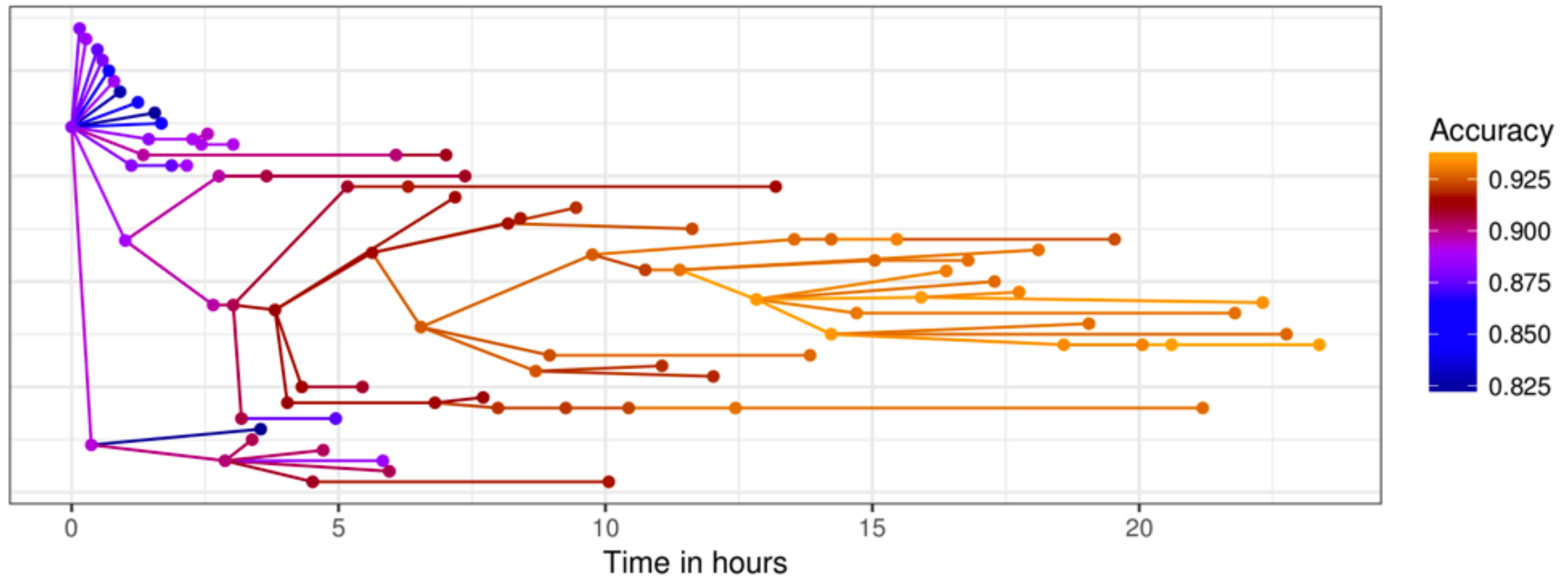




Fusi et al. 2018

GAMA (TU/e): quickly evolves optimal pipelines for a given input dataset

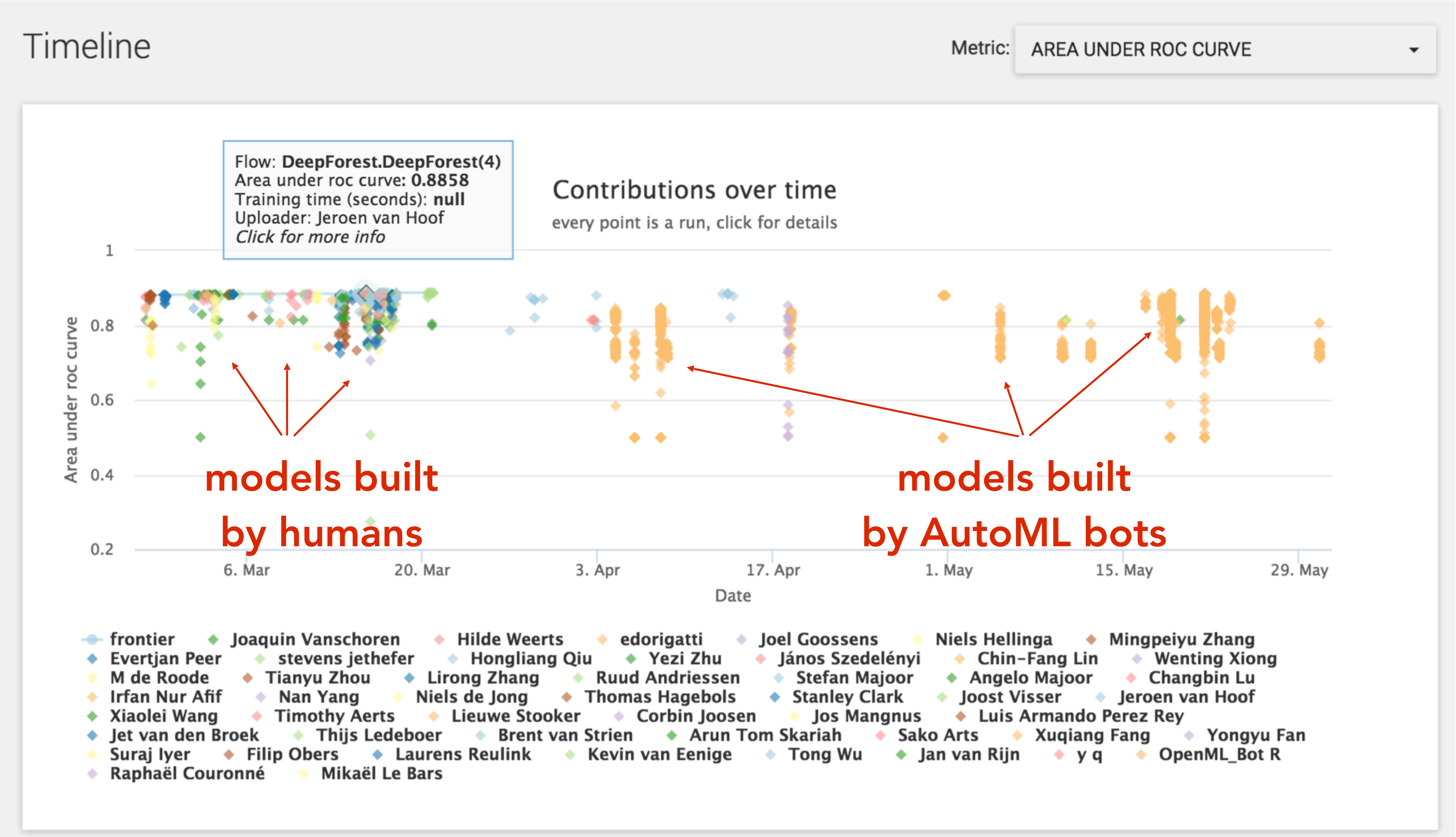




Gijsbers et al. 2018

Human-AI interaction

Algorithms learn from models shared by humans
Humans learn from models built by bots

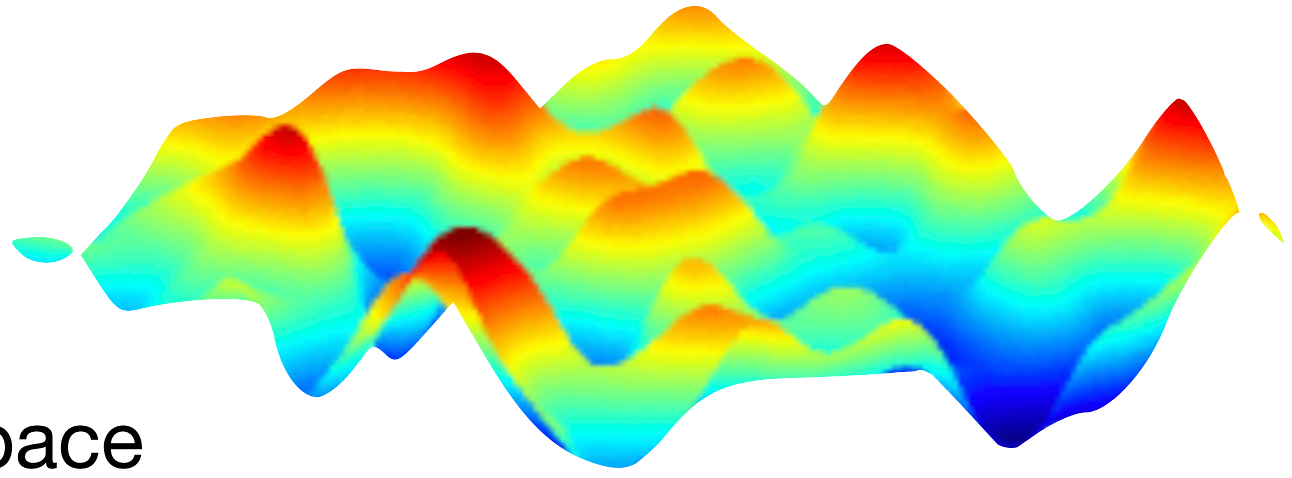


Vanschoren 2018

Meta-learning across tasks: how?

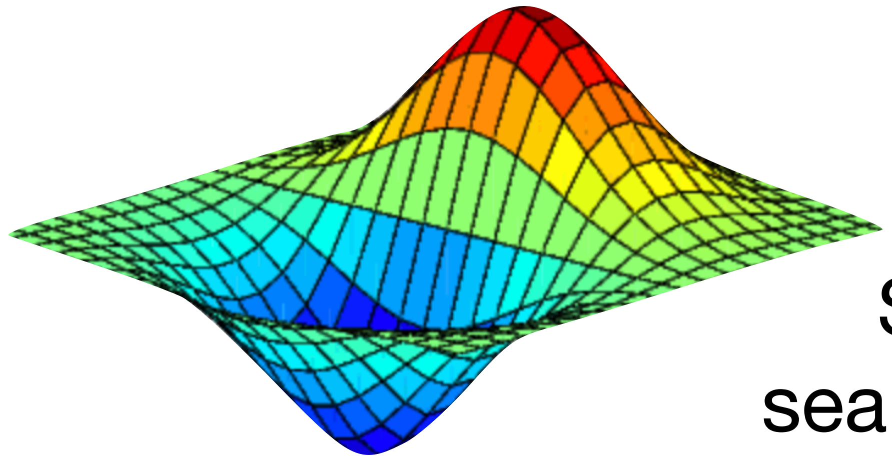
Learning hyperparameter priors

Complex
search space



λ, scores

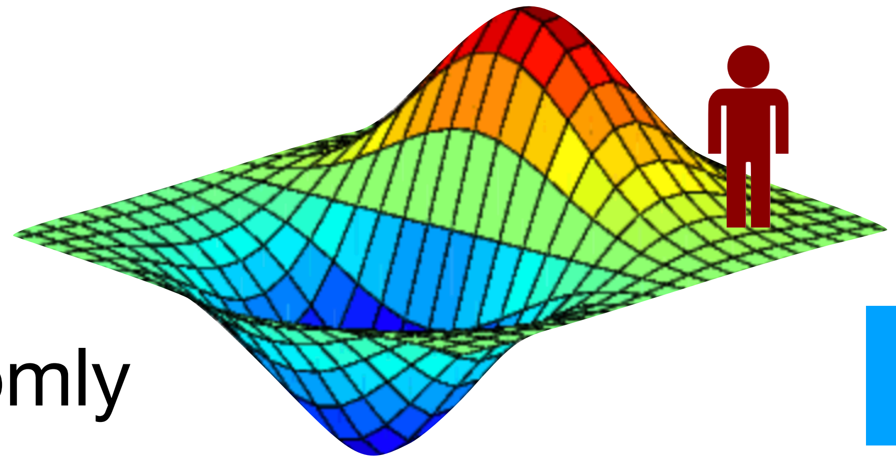
Learner



Simple
search space

Warm starting (what works on similar tasks?)

start randomly

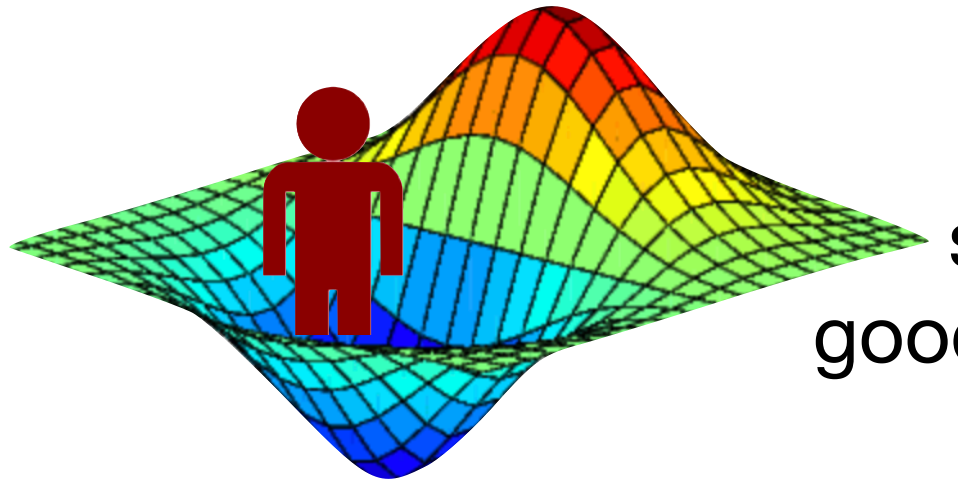


Task

λ, scores

metadata

Learner



start with
good candidates

Meta-models (learn how to build models/components)

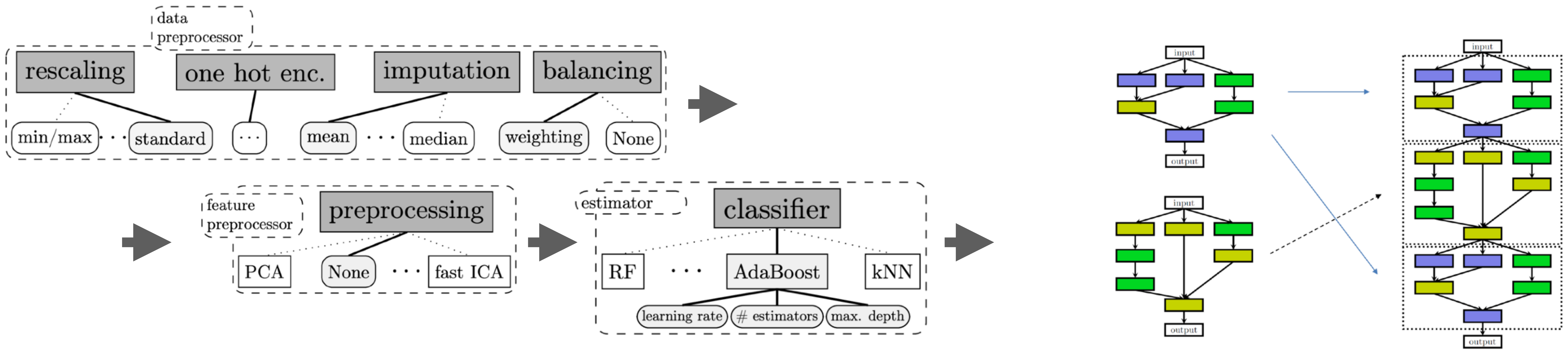
Task Task Task

λ, scores

metadata

Learner

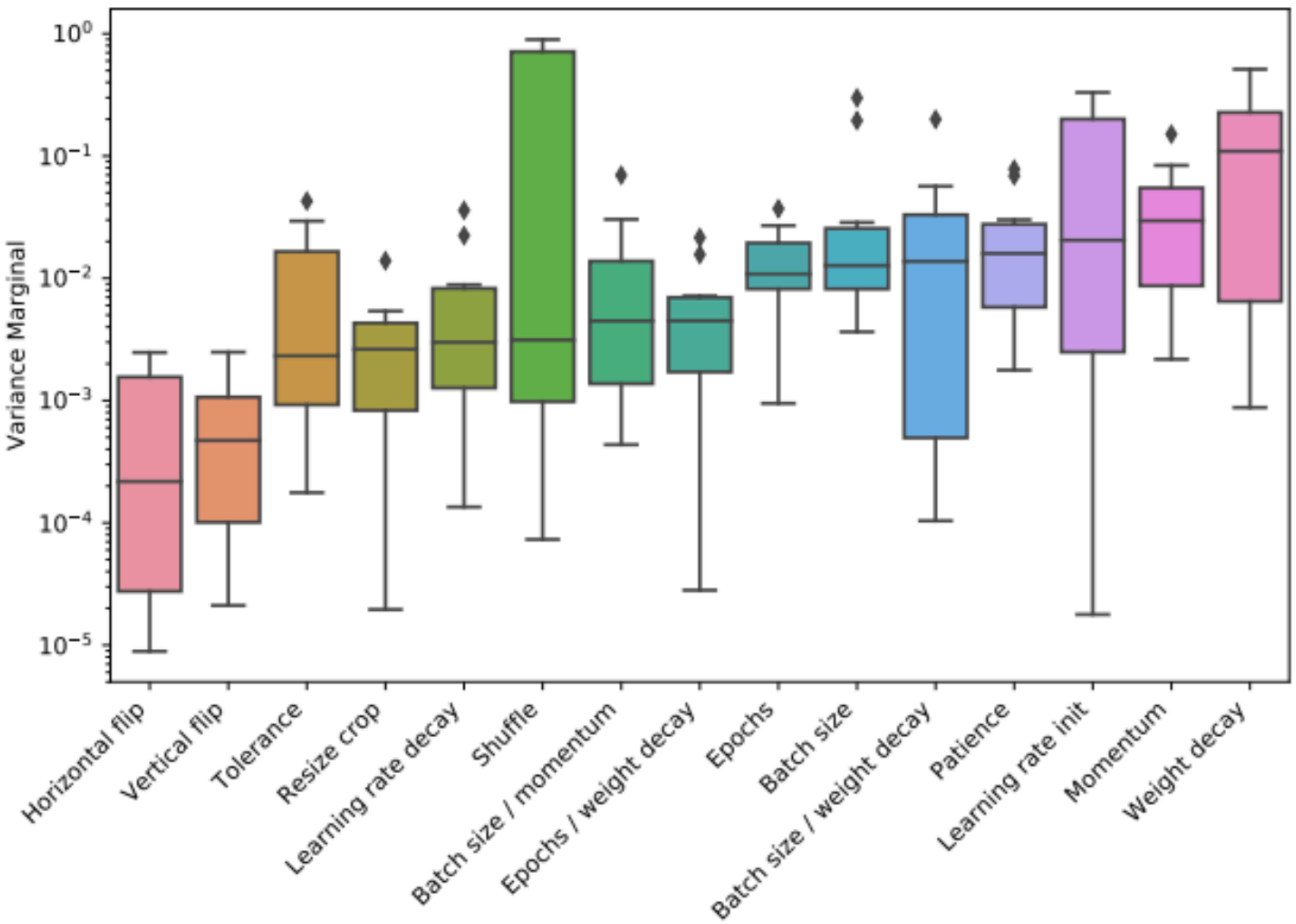
Observation:
current AutoML strongly depends on learned priors



Learn hyperparameter importance

[1] van Rijn & Hutter 2018

- **Functional ANOVA** [1]
  - Select hyperparameters that cause variance in the evaluations.
  - Useful to speed up black-box optimization techniques



ResNets for image classification

Figure source: van Rijn & Hutter, 2018

Learn defaults + hyperparameter importance

[1] Probst et al. 2018

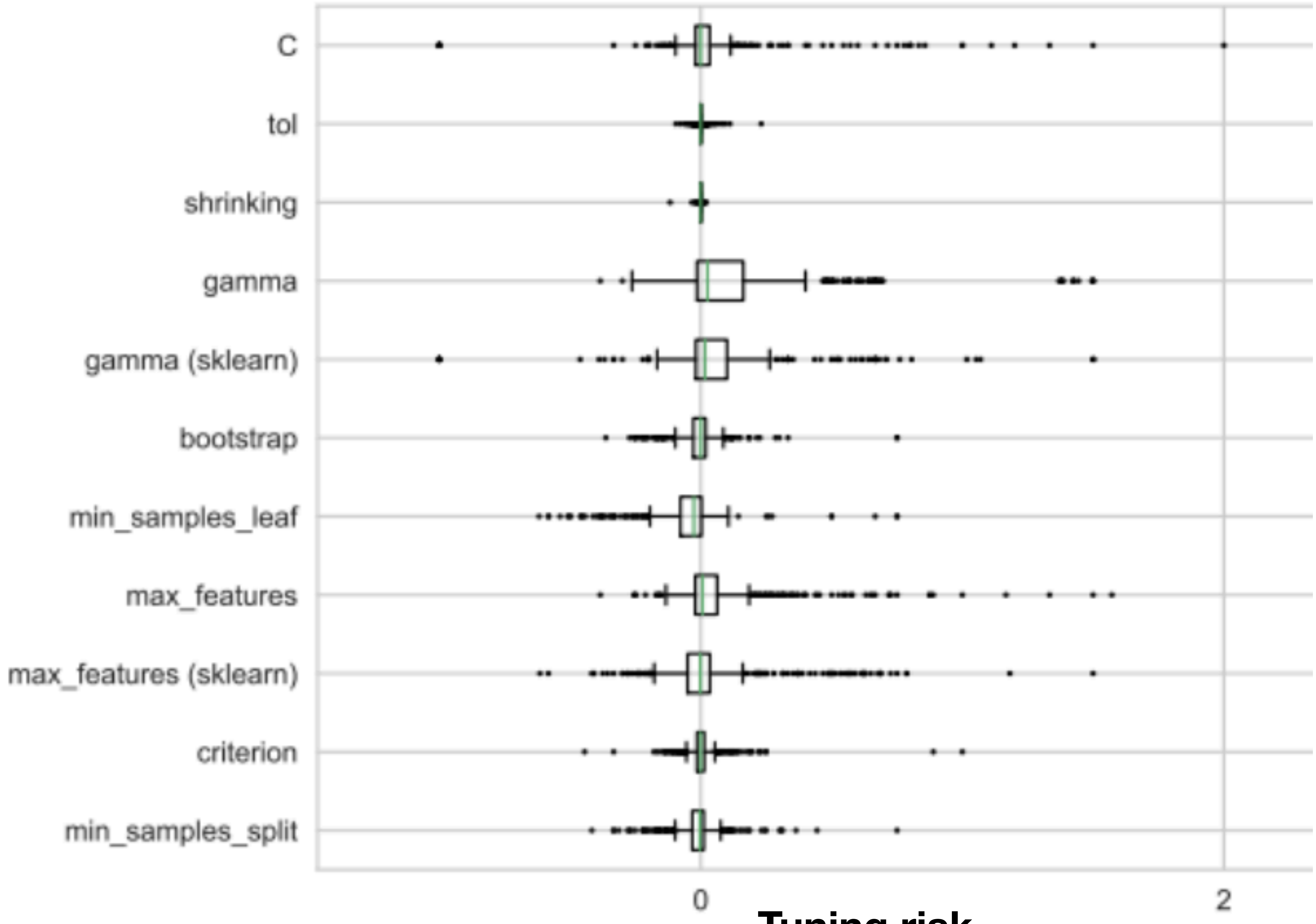
[2] Weerts et al. 2018

[3] van Rijn et al. 2018

- **Tunability** [1,2,3]
  Learn good defaults, measure importance as improvement via tuning

function
max_features
m = 0.16*p
m = p ^0.74
m = 1.15^sqrt(p)
m = sqrt(p)
gamma
m = 0.00574*p
m = 1/p
m = 0.006

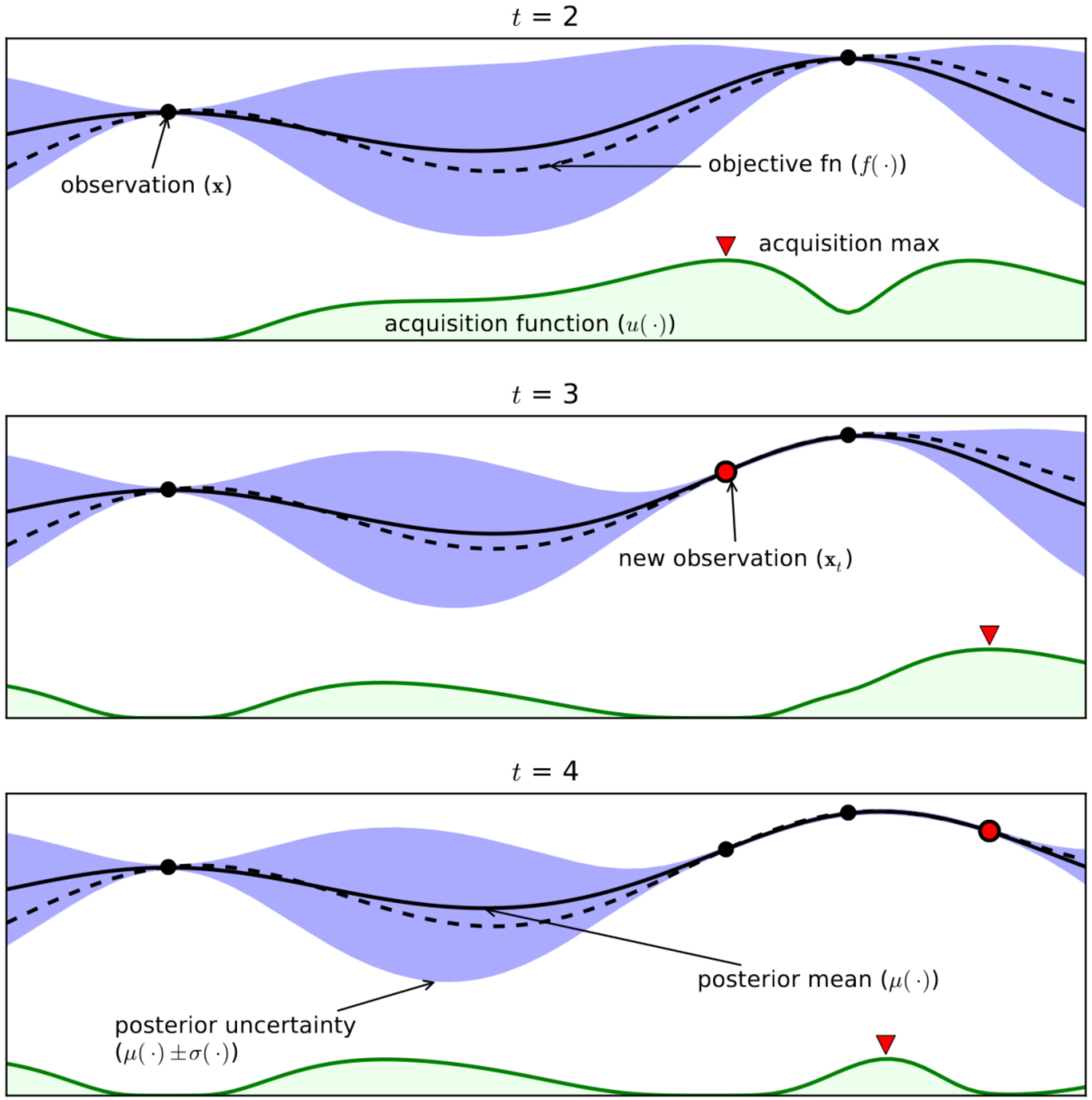
Learned defaults



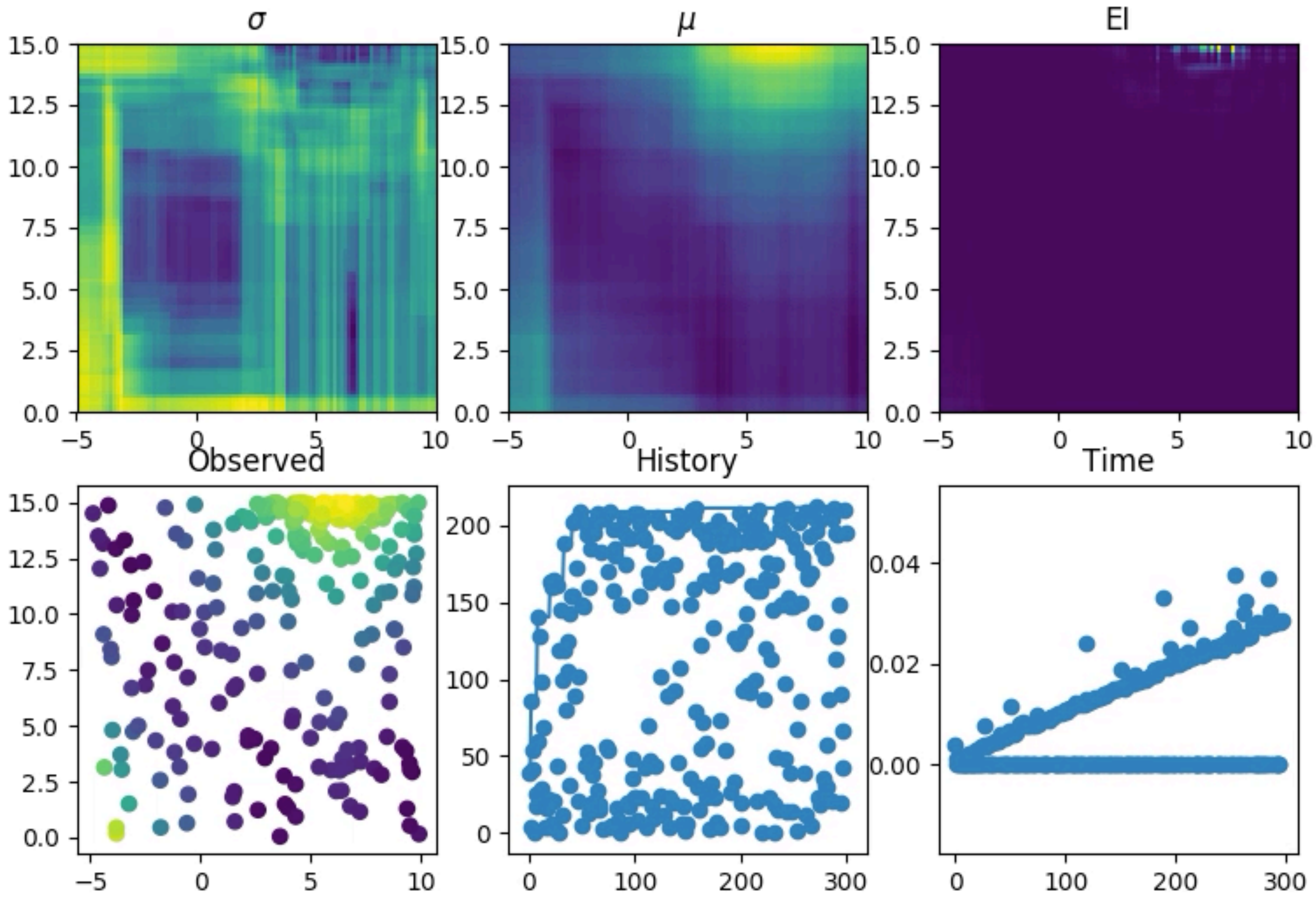
Tuning risk

Bayesian Optimization

- Try initial set of models
- Eg. Those that worked on similar tasks
- Fit a ***surrogate model*** to predict next model
- Use an ***acquisition function*** to trade off exploration and exploitation, e.g. Expected Improvement (EI)
- Used e.g. in AlphaGo



Optimization: Bayesian Optimization

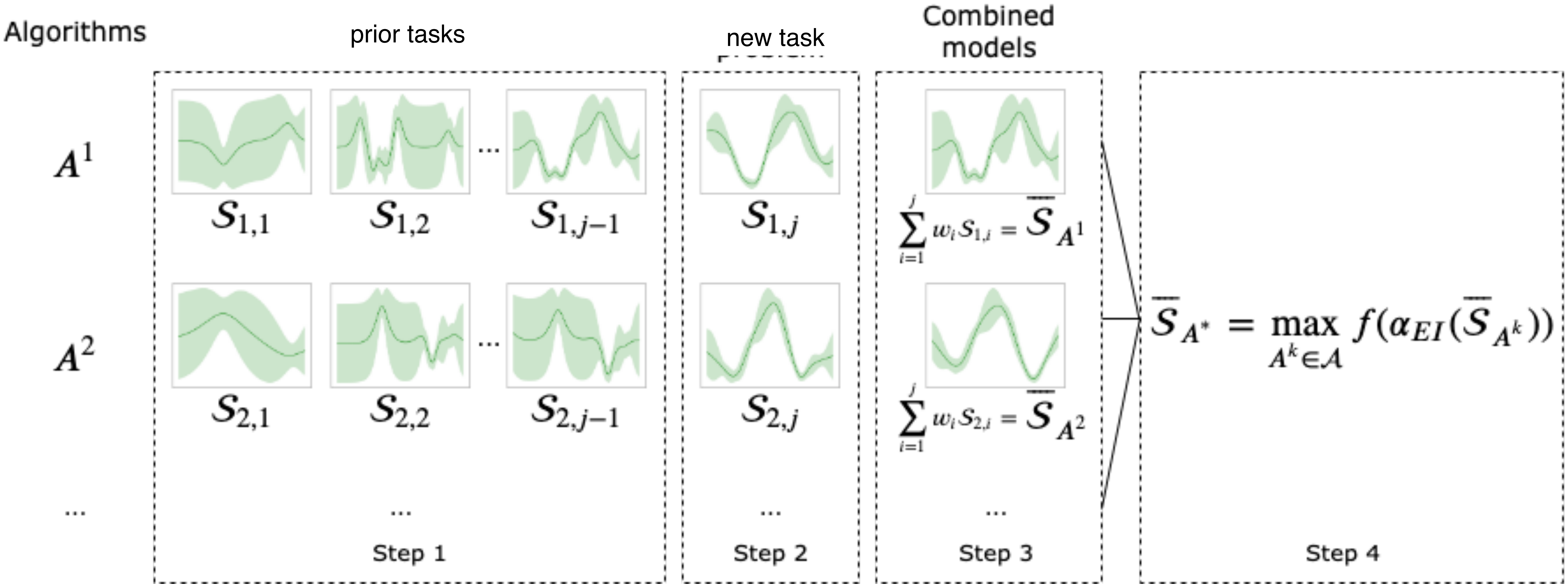


animation by Jeroen van Hoof

Manolache & Vanschoren 2019

Surrogate model transfer

- If task j is similar to the new task, its surrogate model S_j will likely transfer well
- Sum up all S_j predictions, weighted by task similarity
- Build combined surrogate, weighted by current performance on new task[2]

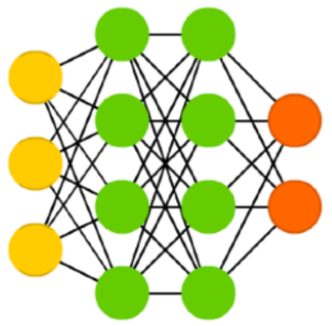


Learn basis expansions for hyperparameters

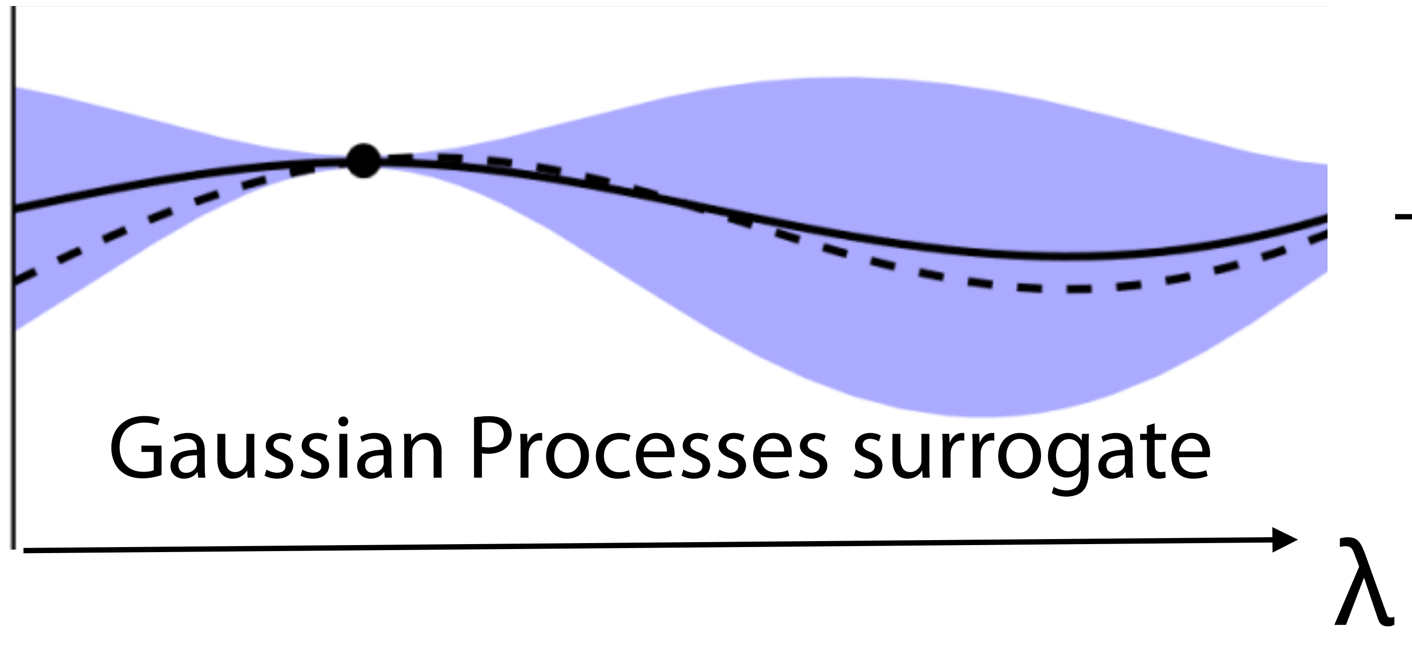
- Hyperparameters can interact in very non-linear ways
- Use a neural net to learn a suitable transform $\phi_z(\lambda)$ so that they behave linearly
- Used in SageMaker AutoML

Learn basis expansion on lots of data (e.g. OpenML)

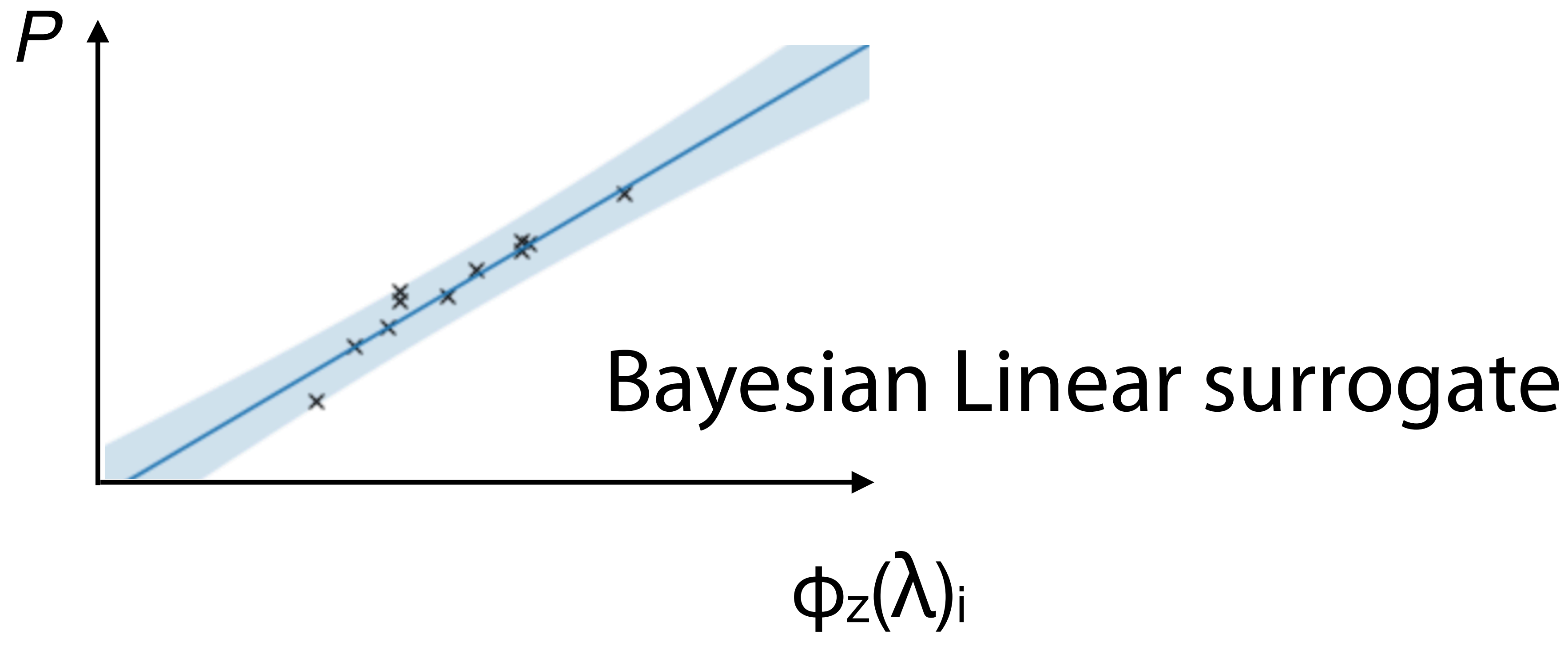
λ_i, score →



→ $\phi_z(\lambda)$

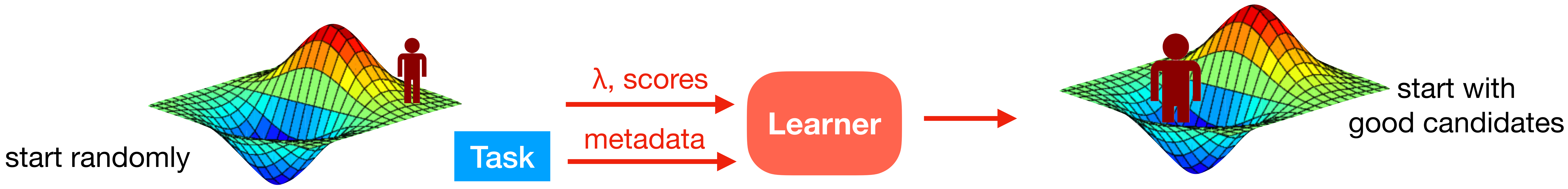


→ $\phi_z(\lambda)$ →



Warm starting

(what works on similar tasks?)



How to measure task similarity?

- Hand-designed (statistical) meta-features that describe (tabular) datasets [1]
- Task2Vec: task embedding for image data [2]
- Optimal transport: similarity measure based on comparing probability distributions [3]
- Metadata embedding based on textual dataset description [4]
- Dataset2Vec: compares batches of datasets [5]
- Distribution-based invariant deep networks [6]

Figure source: Alvarez-Melis et al. 2020

[1] Vanschoren 2018
[2] Achille et al. 2019
[3] Alvarez-Melis et al. 2020
[4] Drori et al. 2019
[5] Jooma et al. 2020
[6] de Bie et al. 2020

Feurer et al. 2015

Warm-starting with kNN

- Find k most similar tasks, warm-start search with best λ_i
    - Auto-sklearn: Bayesian optimization (SMAC)
        - Meta-learning yield better models, faster
        - Winner of several AutoML Challenges

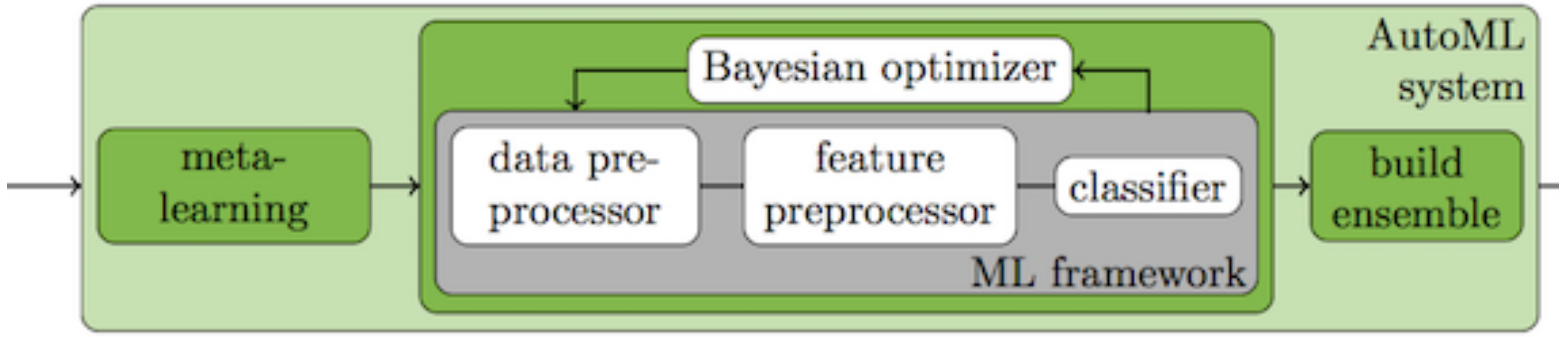
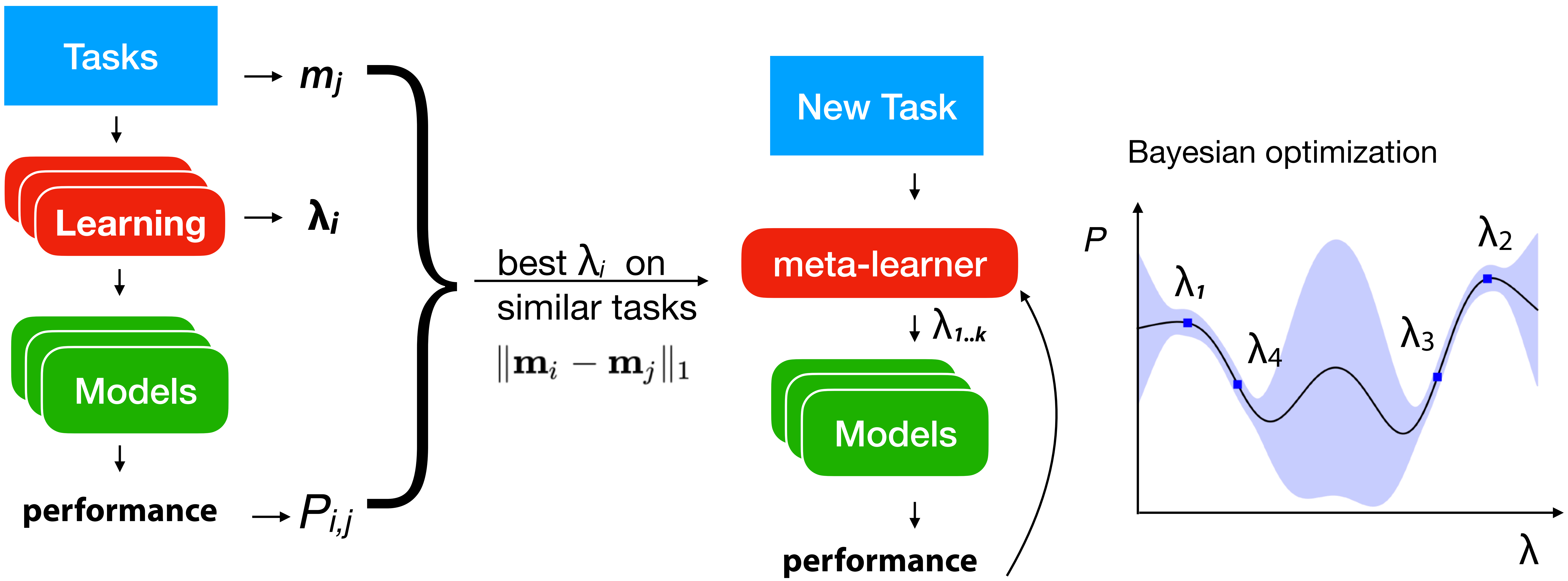


Figure source: Feurer et al., 2015

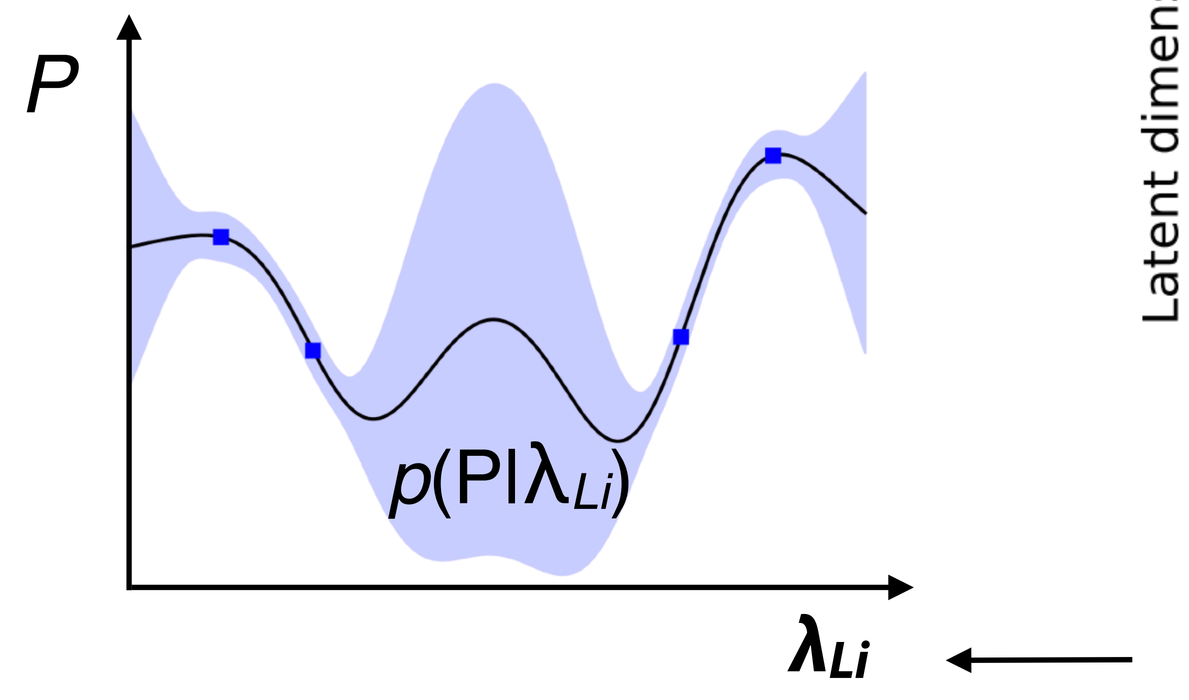


[Fusi et al. 2017]

Probabilistic Matrix Factorization

- Collaborative filtering: configurations λ_i are `rated' by tasks t_j

- Learn latent representation for tasks T and configurations λ
- Use meta-features to warm-start on new task
- Returns probabilistic predictions for Bayesian optitmization
- Used in Azure AutoML



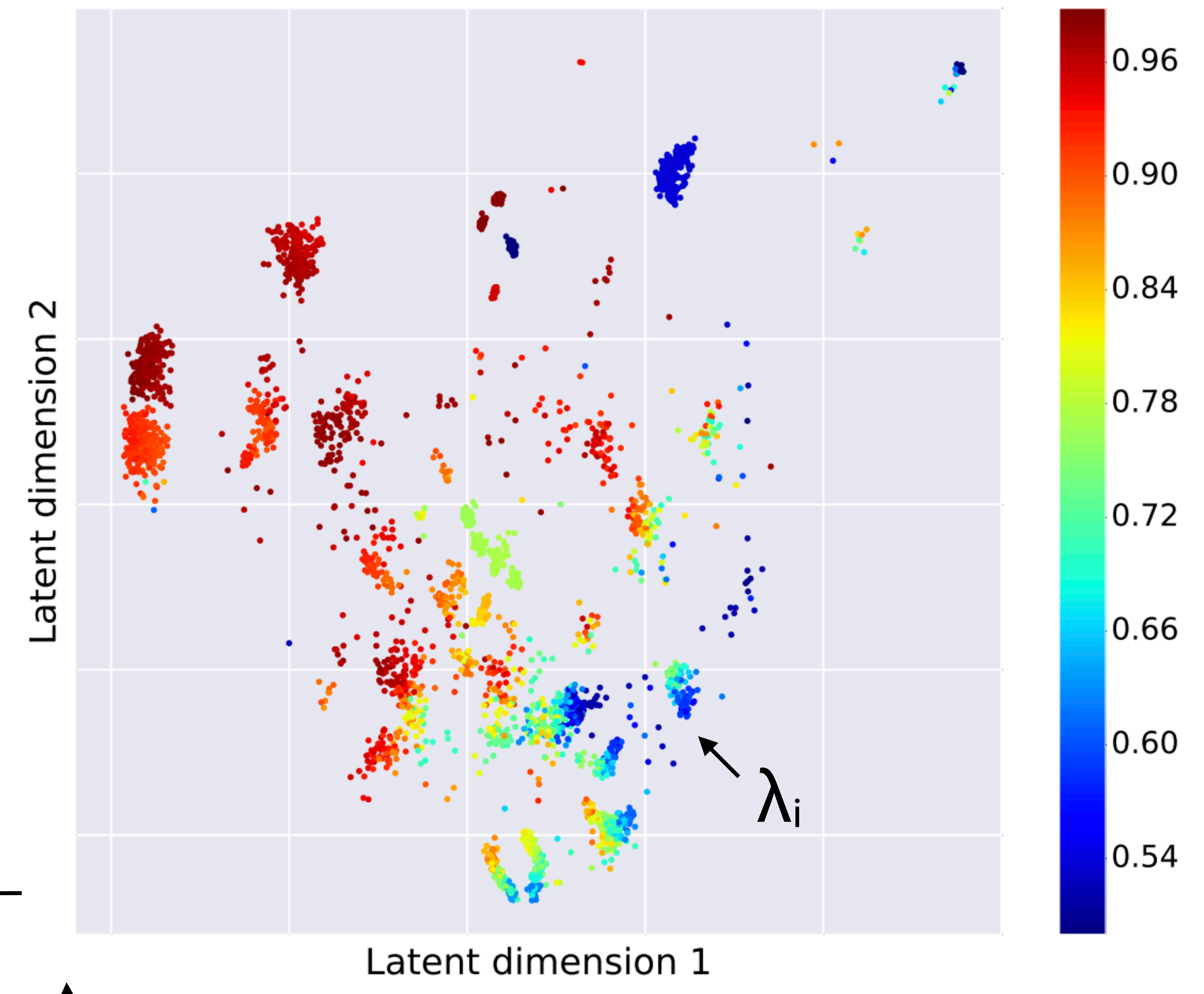
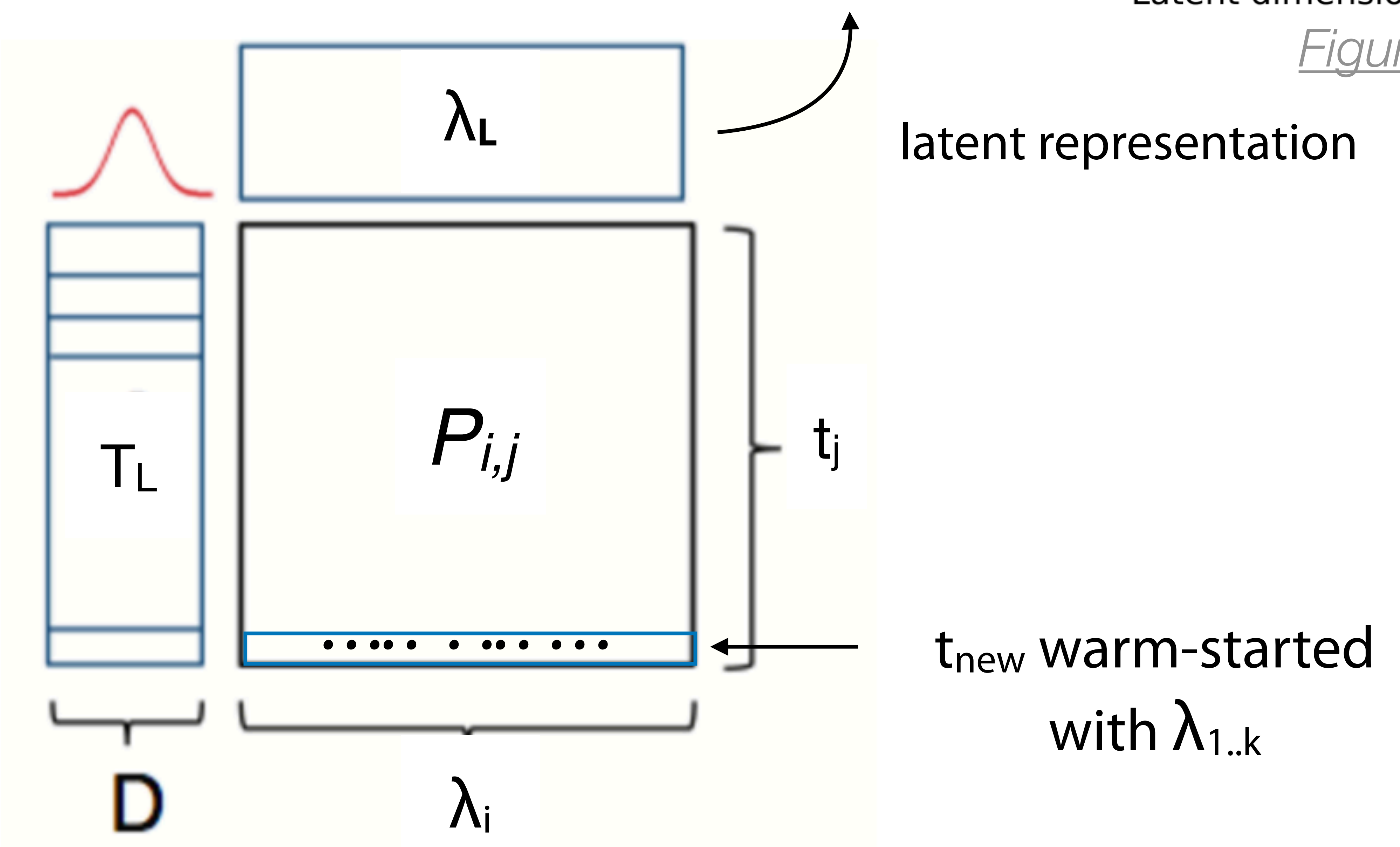
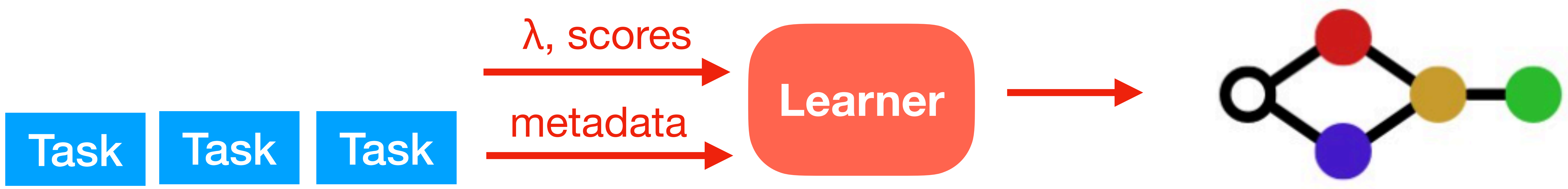


Figure source: Fusi et al., 2017



Meta-models

(learn how to build models/components)



[1] Brazdil et al. 2009, Lemke et al. 2015

[2] Sun and Pfahringer 2013, Pinto et al. 2017

[3] Sanders and C. Giraud-Carrier 2017

[4] Yang et al. 2018

Algorithm selection models

- Learn direct mapping between meta-features and $P_{i,j}$
    - Zero-shot meta-models: predict best λ_i given meta-features [1]

    $$\boldsymbol{m_j} \longrightarrow \boxed{\text{meta-learner}} \longrightarrow \lambda_{best}$$

    - Ranking models: return ranking $\lambda_{1..k}$ [2]

    $$\boldsymbol{m_j} \longrightarrow \boxed{\text{meta-learner}} \longrightarrow \lambda_{1..k}$$

    - Predict which algorithms / configurations to consider / tune [3]

    $$\boldsymbol{m_j} \longrightarrow \boxed{\text{meta-learner}} \longrightarrow \boldsymbol{\Lambda}$$

    - Predict performance / runtime for given Θ_i and task [4]

    $$\boldsymbol{m_j}, \lambda_i \longrightarrow \boxed{\text{meta-learner}} \longrightarrow P_{ij}$$

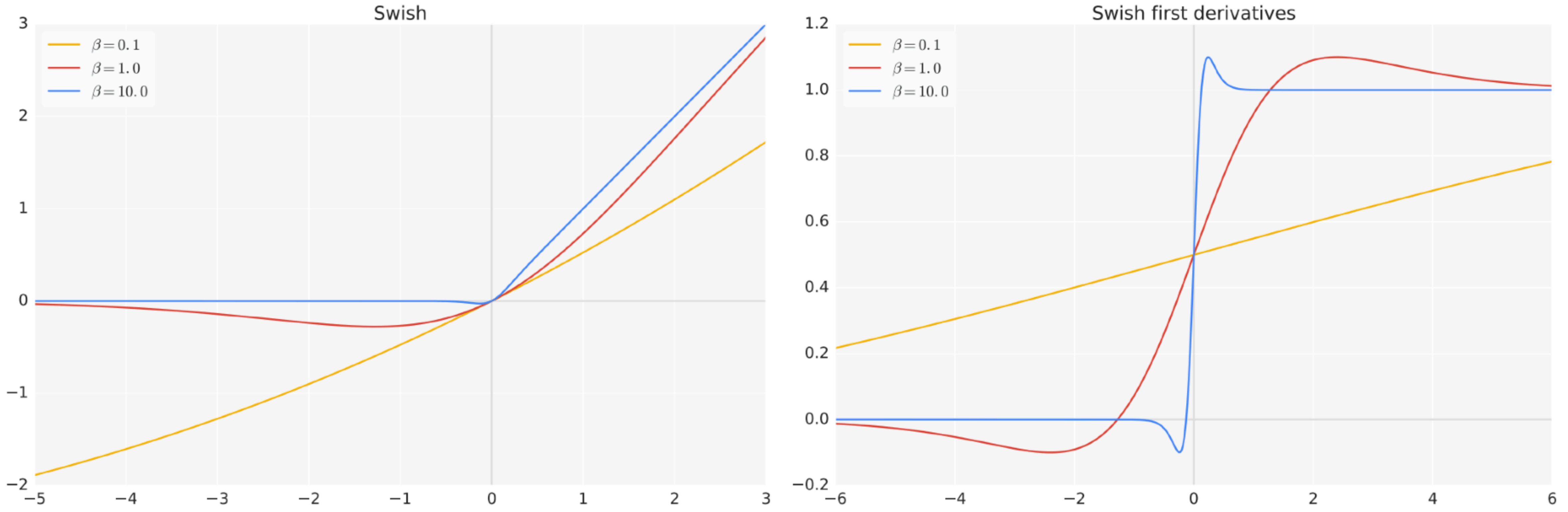
- Can be integrated in larger AutoML systems: warm start, guide search,…

Learning model components

[1] Ramachandran et al. 2017
[2] Bello et al. 2017
[3] Volpp et al. 2020

- Learn nonlinearities: RL-based search of space of likely useful activation functions [1]
  - E.g. Swish can outperform ReLU

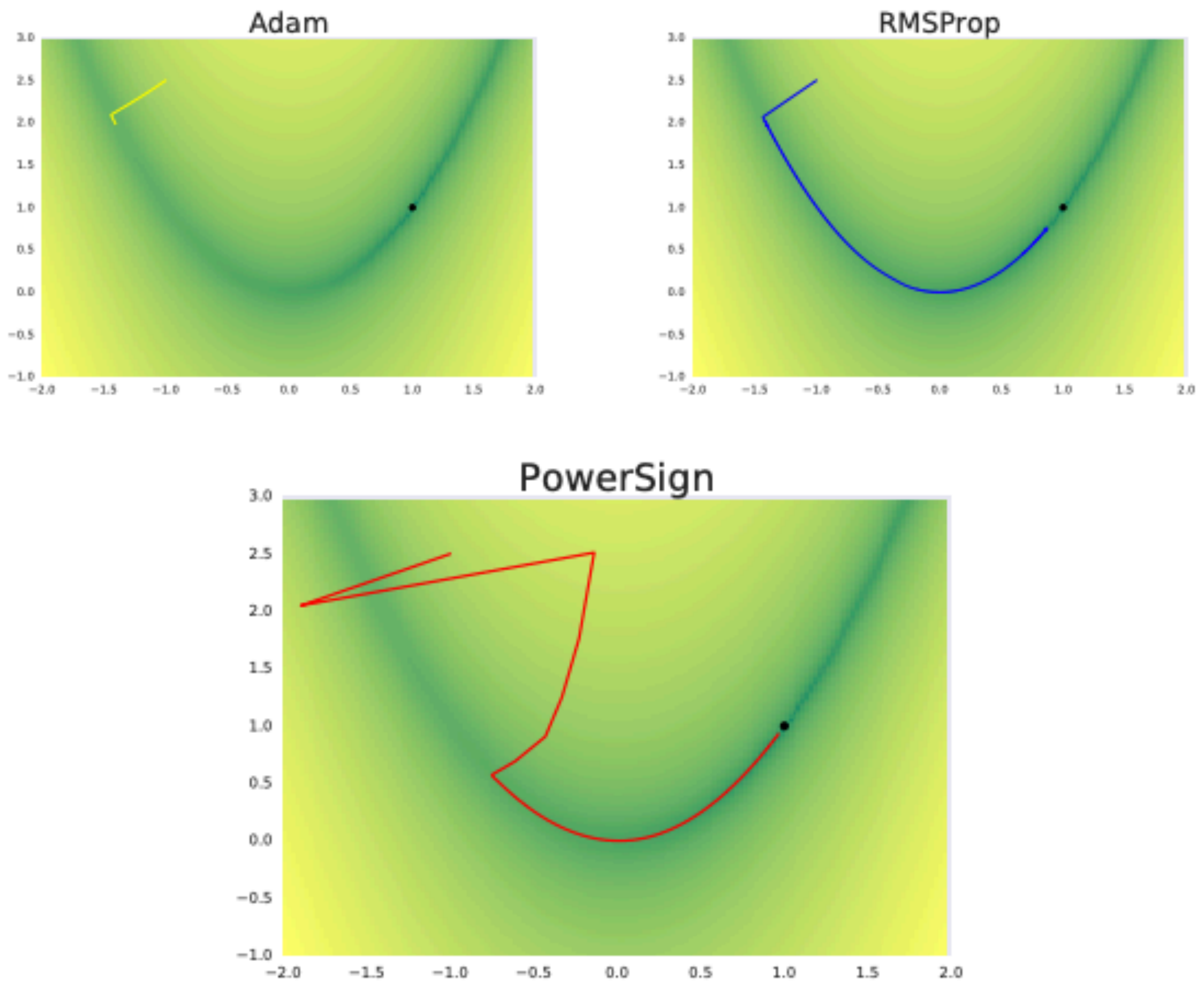
$$Swish : \frac{x}{1 + e^{-\beta x}}$$



- Learn optimizers: RL-based search of space of likely useful update rules [2]
  - E.g. PowerSign can outperform Adam, RMPprop

$$PowerSign : e^{sign(g)sign(m)}g$$

g: gradient, m:moving average



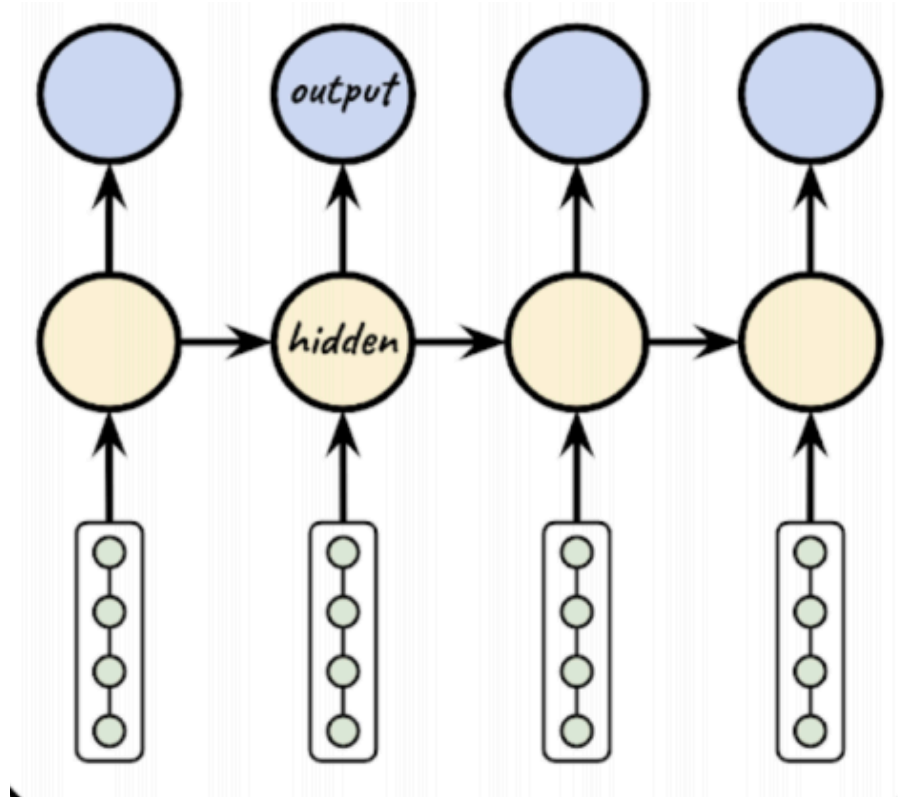
- Learn acquisition functions for Bayesian optimization [3]

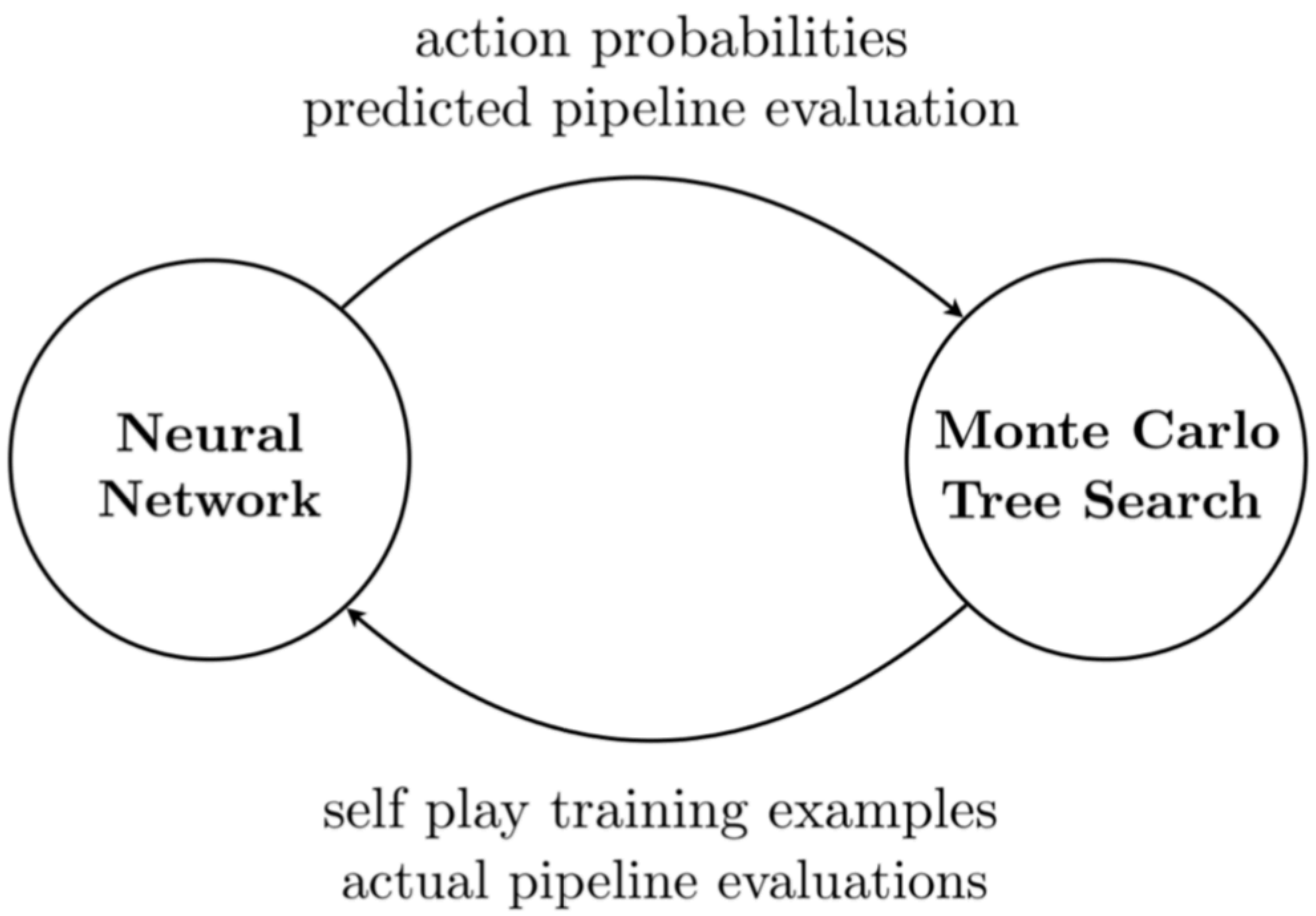
Figure source: Ramachandran et al., 2017 (top), Bello et al. 2017 (bottom)

Monte Carlo Tree Search + reinforcement learning

MOSAIC [Rakotoarison et al. 2019]
AlphaD3M [Drori et al. 2019]

- ***Self-play:***
  - Game actions: insert, delete, replace components in a pipeline
  - Monte Carlo Tree Search builds pipelines given action probabilities
  - Neural network (LSTM) Predicts pipeline performance





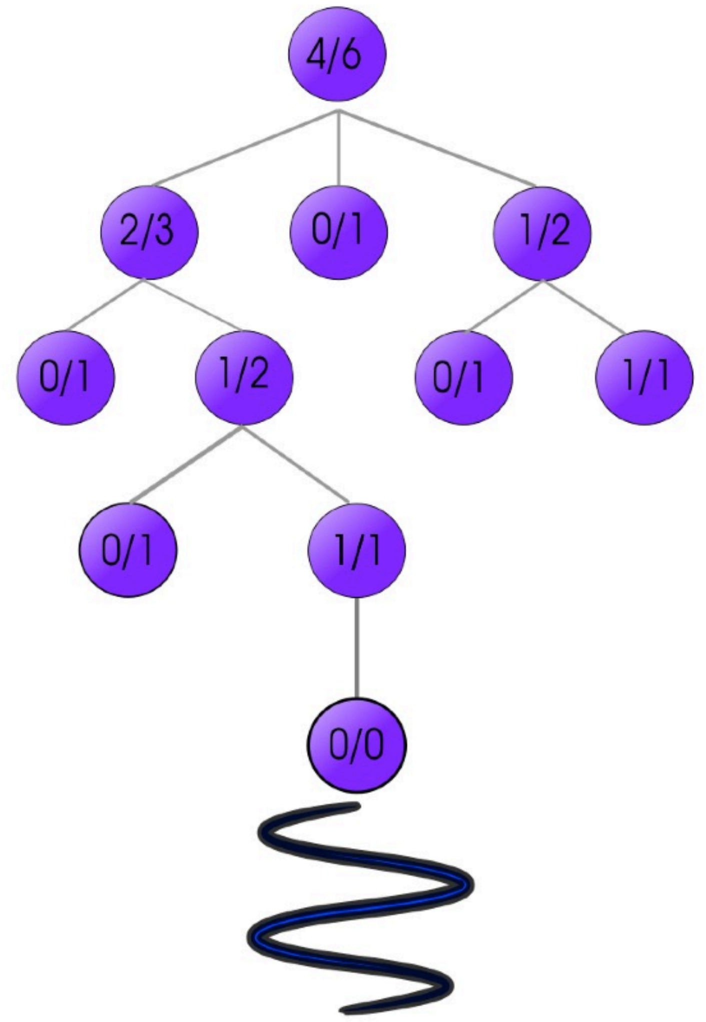


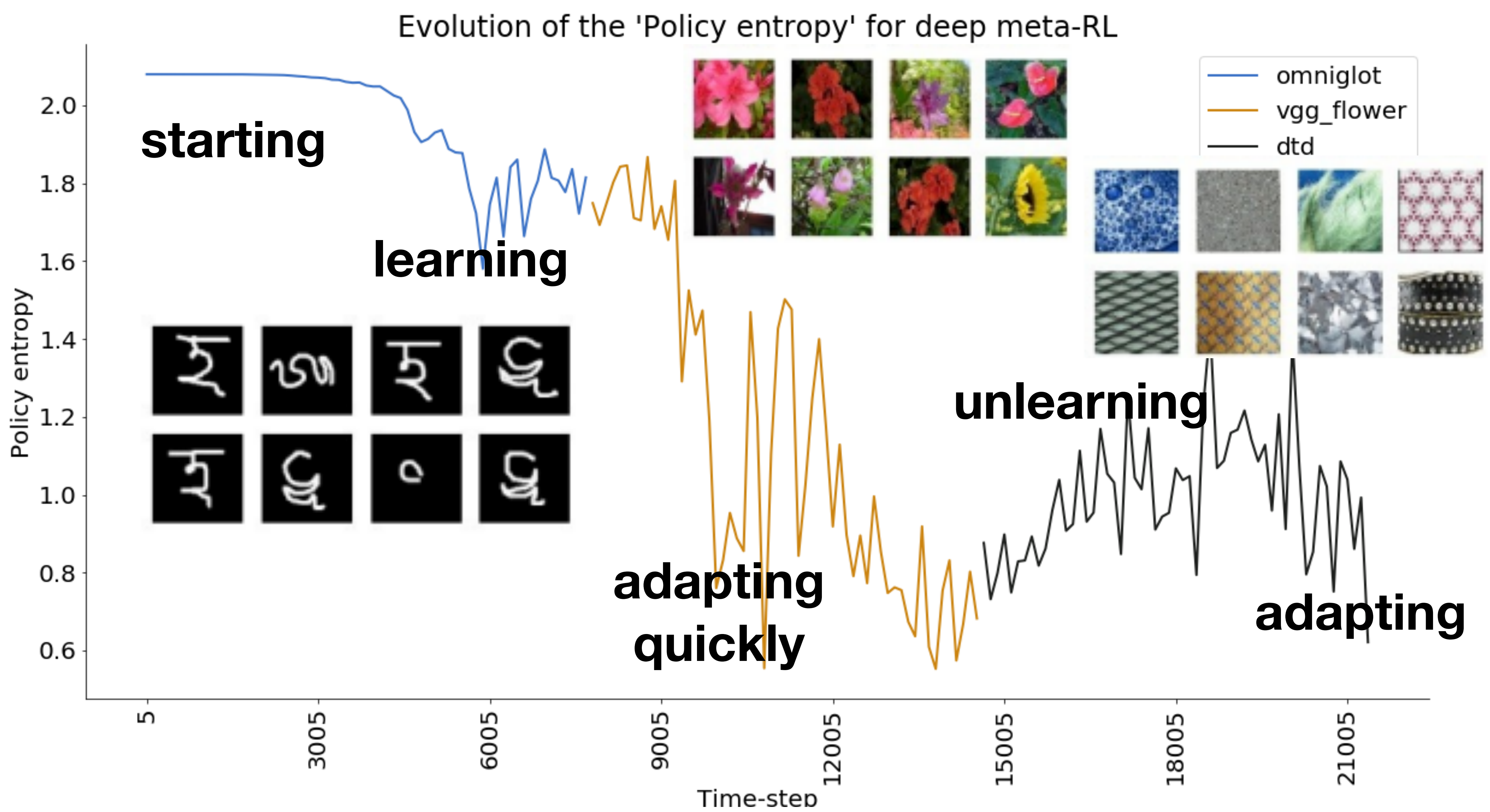
Figure source: Drori et al., 2019

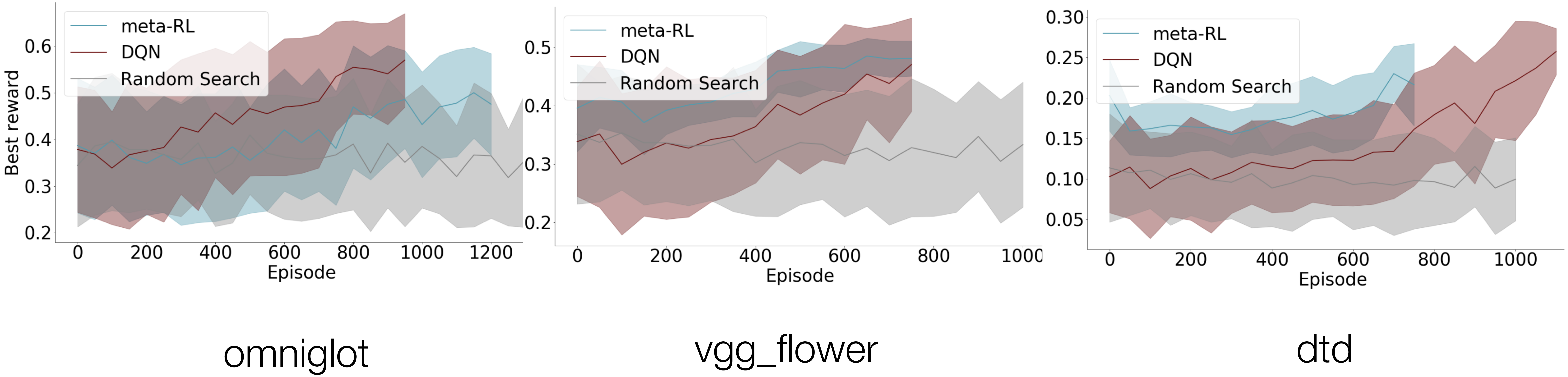
Gomez & Vanschoren, 2019

Meta-Reinforcement Learning for NAS

Image classifiers for increasingly difficult tasks:

- Initially slower than normal RL techniques, but faster after a few tasks
- Policy entropy (agent predictability) shows learning, forgetting, re-learning,….

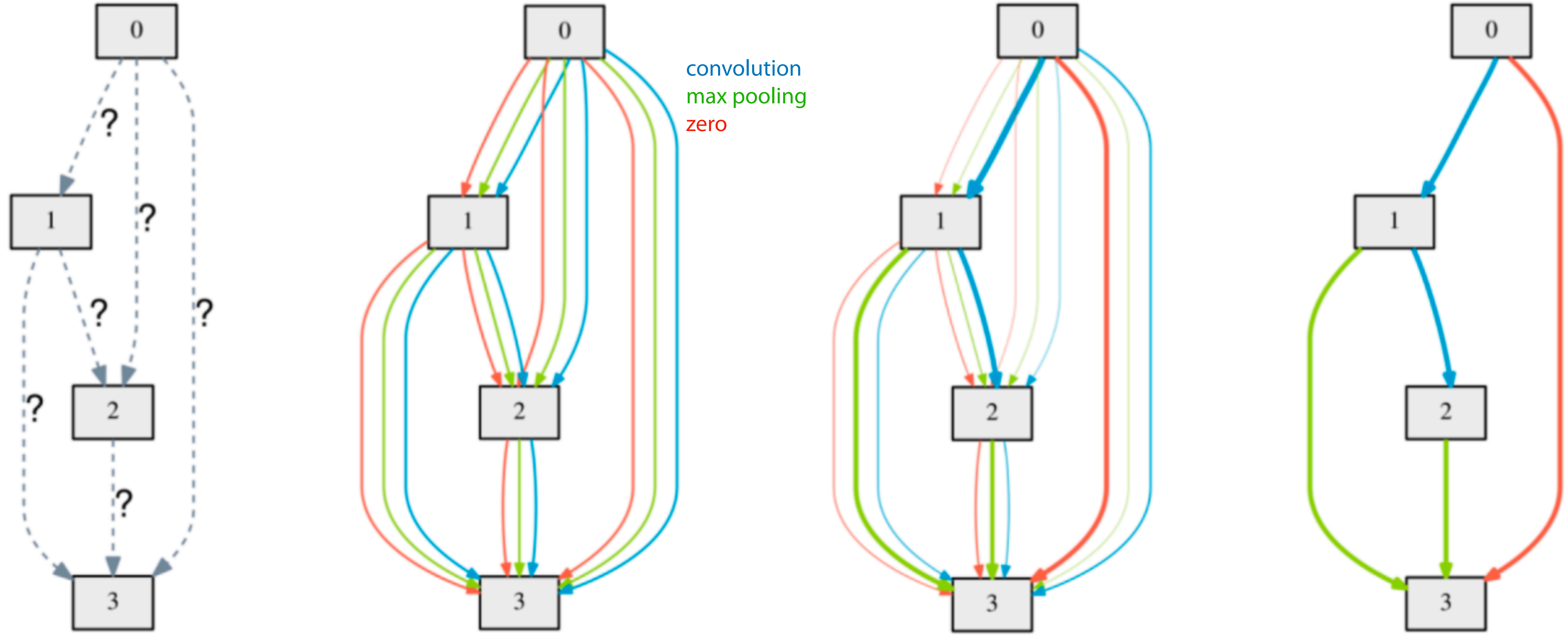




DARTS

Liu et al. 2018

- Fixed backbone, each edge can be any operation (discrete or continuous)
- All operators get weight α_i -> *continuous, differentiable search space*
- Efficient: tune architecture and model weights at the same time (weight sharing)
- Compositionality: learn smaller cells and repeat them in macro-architecture



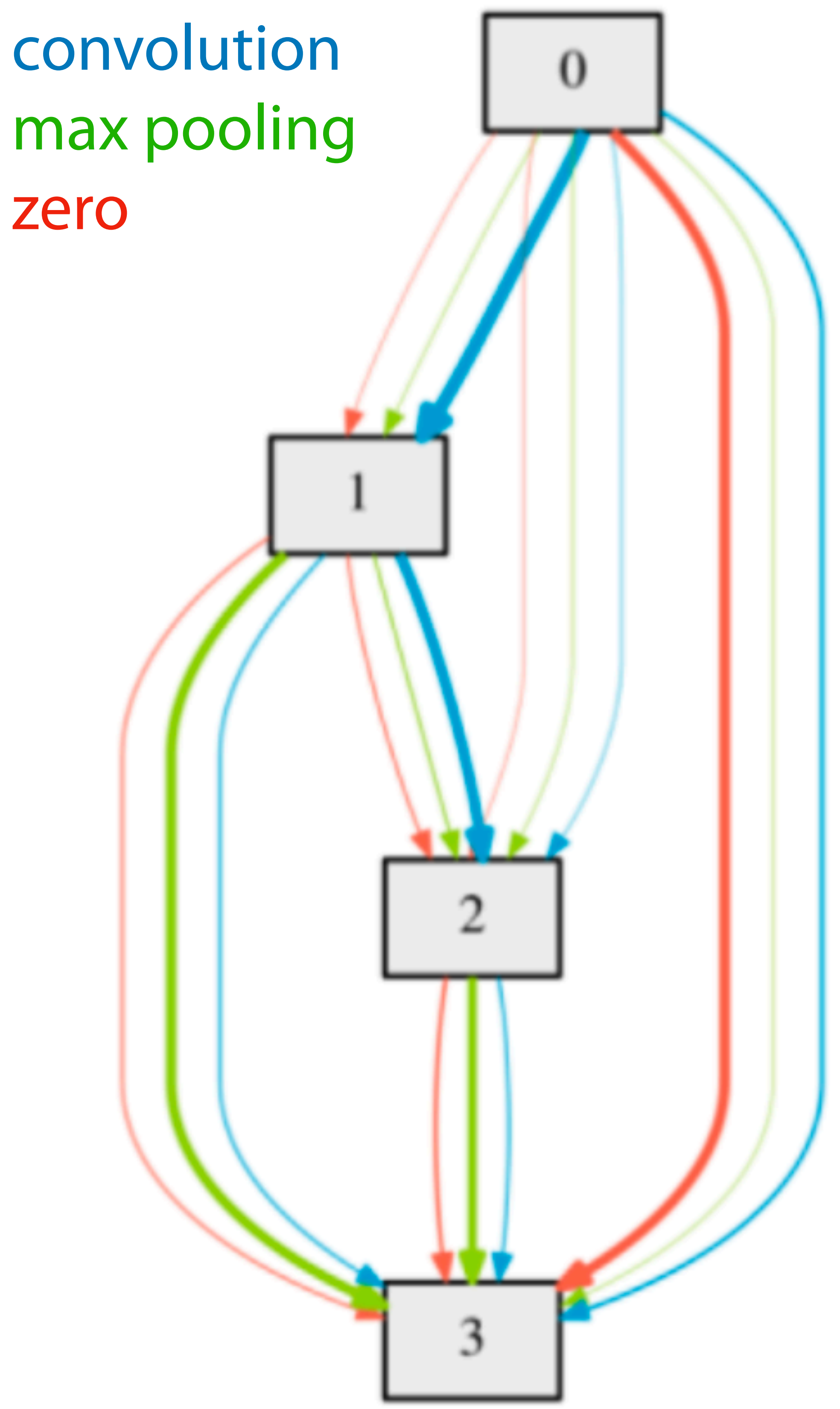
One-shot model **operator weights α_i** **interleaved optimization of α_i and ω_j with SGD** **argmax α_i**

MetaNAS: meta-learning + NAS

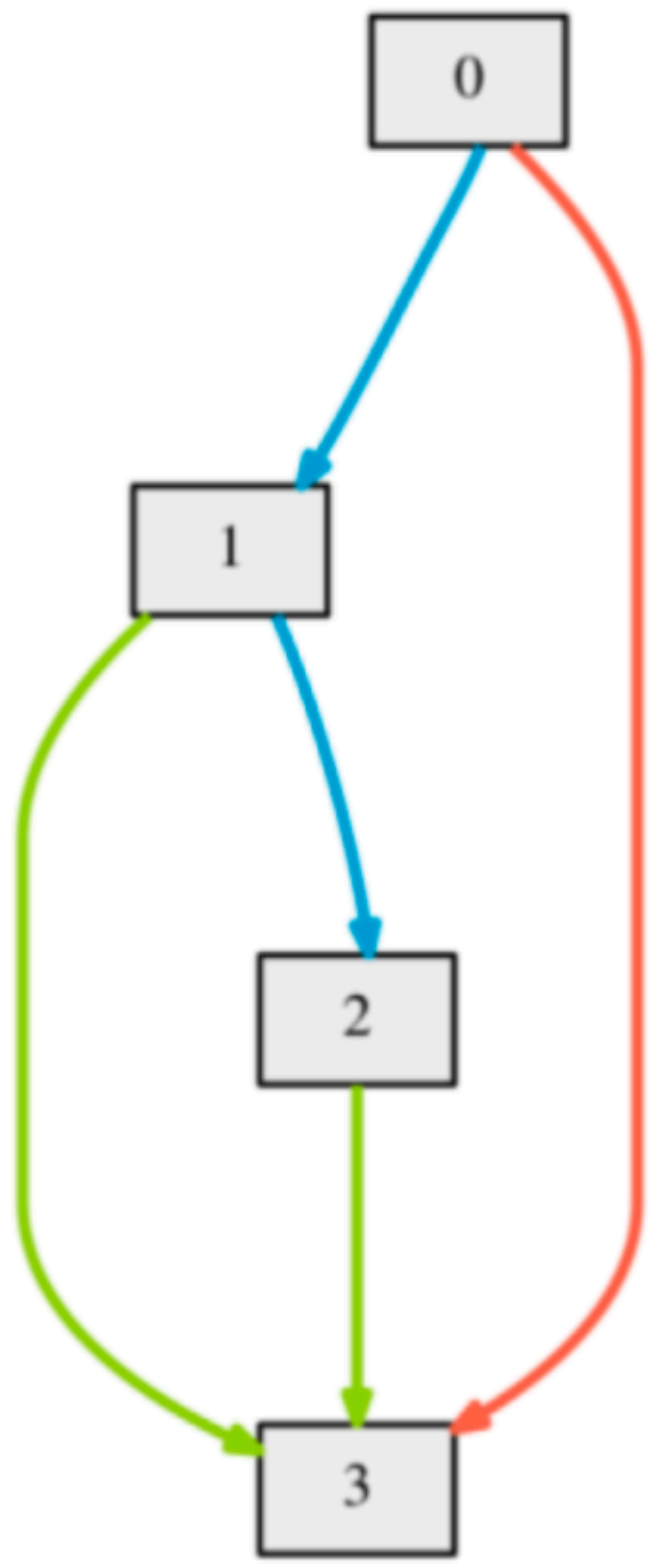
Elsken et al., 2020

- Use meta-learning (MAML) to learn a good weight initialization for the network

Meta-learn initial operator weights α_i from previous tasks



interleaved optimization of α_i and ω_j with SGD

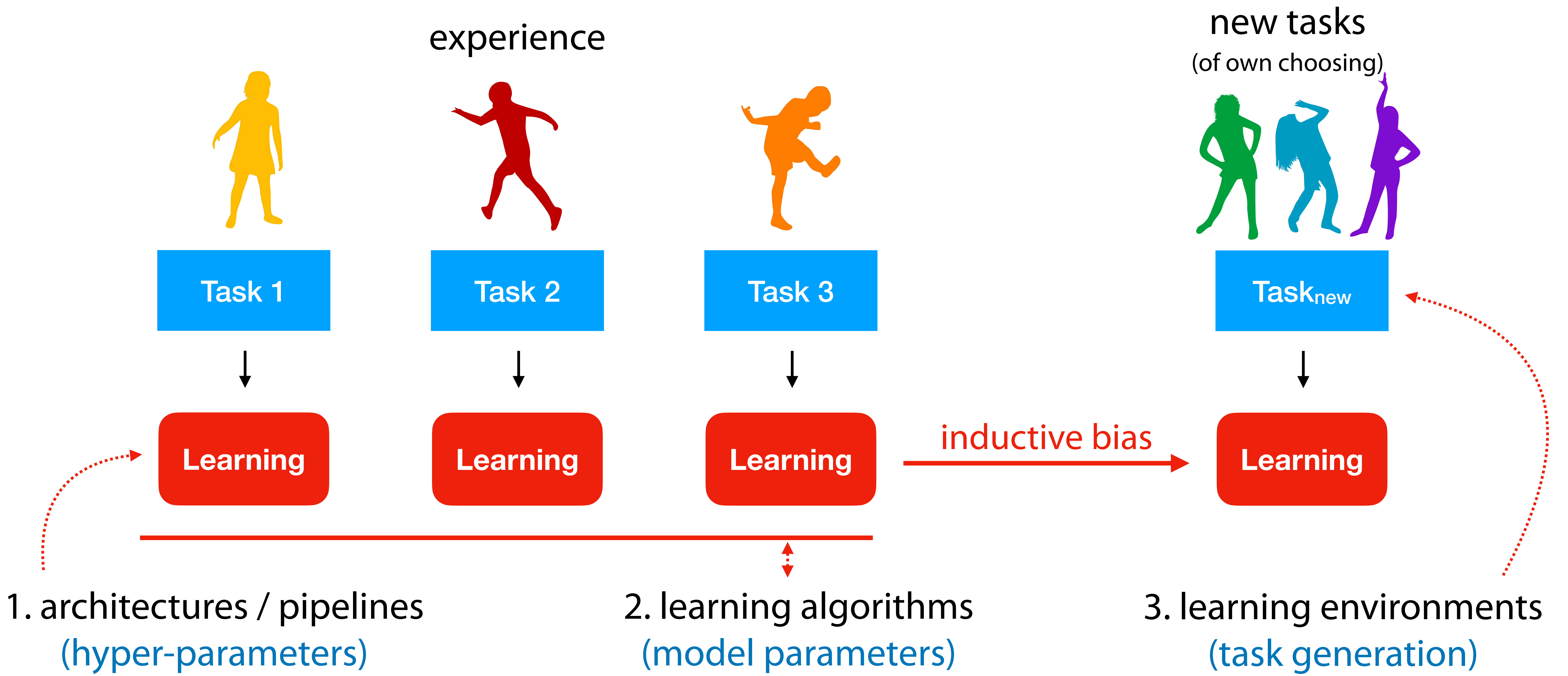


argmax α_i

Clune 2019

What can we learn to learn?

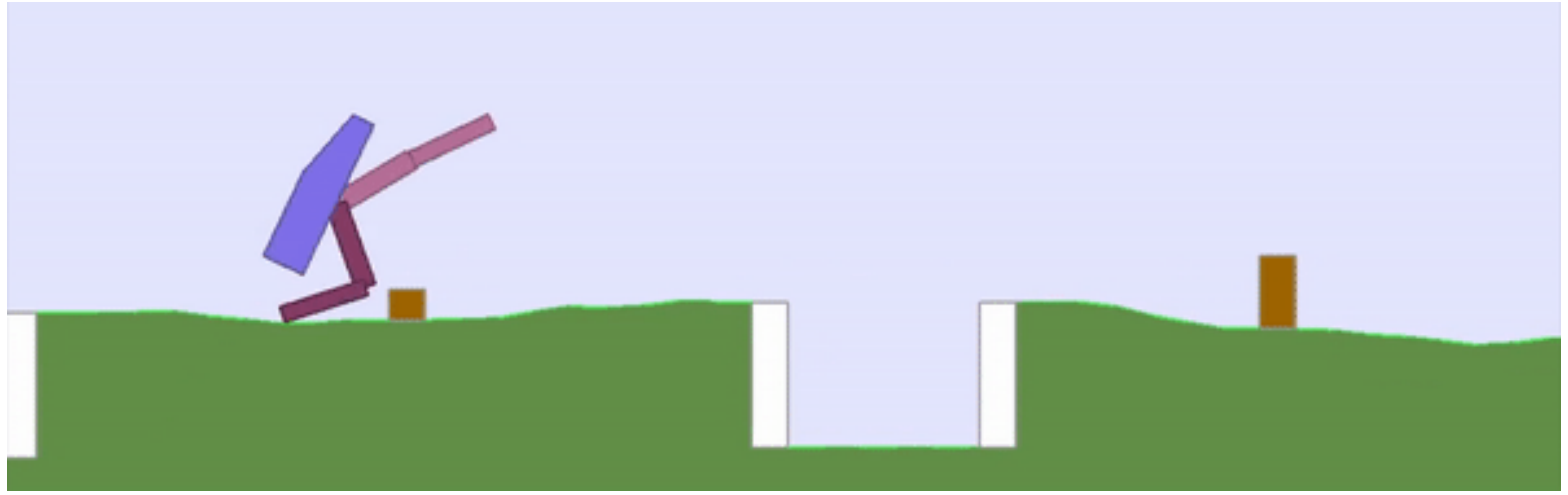
3 pillars



Training Task Acquisition

Wang et al. 2019
Wang et al. 2020

- Ultimately, meta-learning translates constraints on the learner to constraints on the data
  - The biases we don't put in manually have to be learnable from data
- Can we automatically create new tasks to inform and challenge our meta-learners?
- Paired open-ended trailblazer (POET): evolves a parameterized environment Θ_E for agent Θ_A
  - Select agents that can solve challenges AND evolve environments so they are solvable



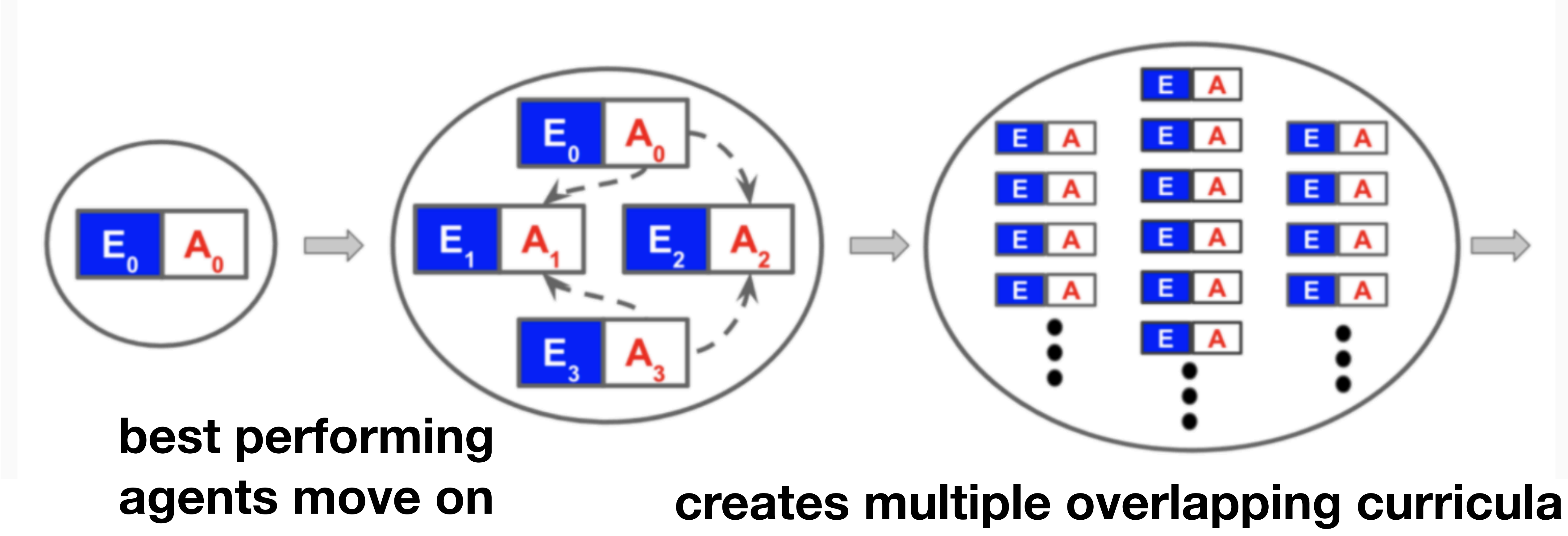
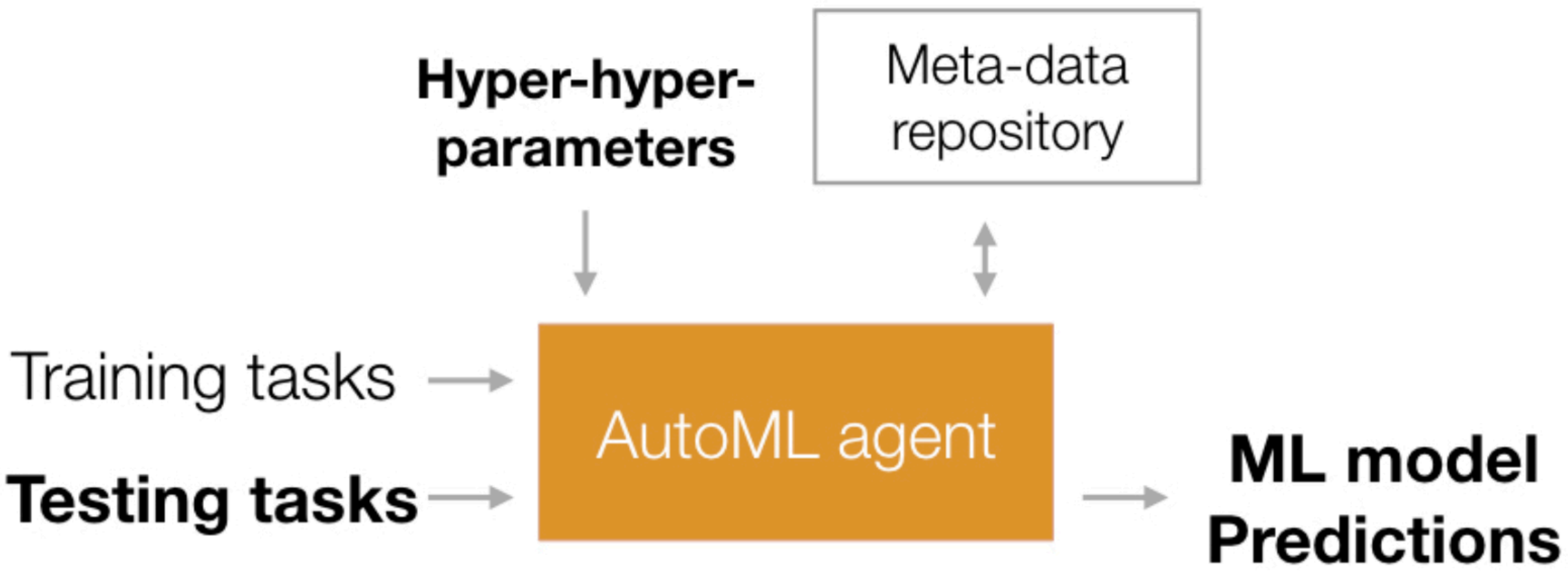


Figure source: Wang et al. 2019

Does POET scale? Increasingly difficult 3D terrain, 18 degrees of freedom.

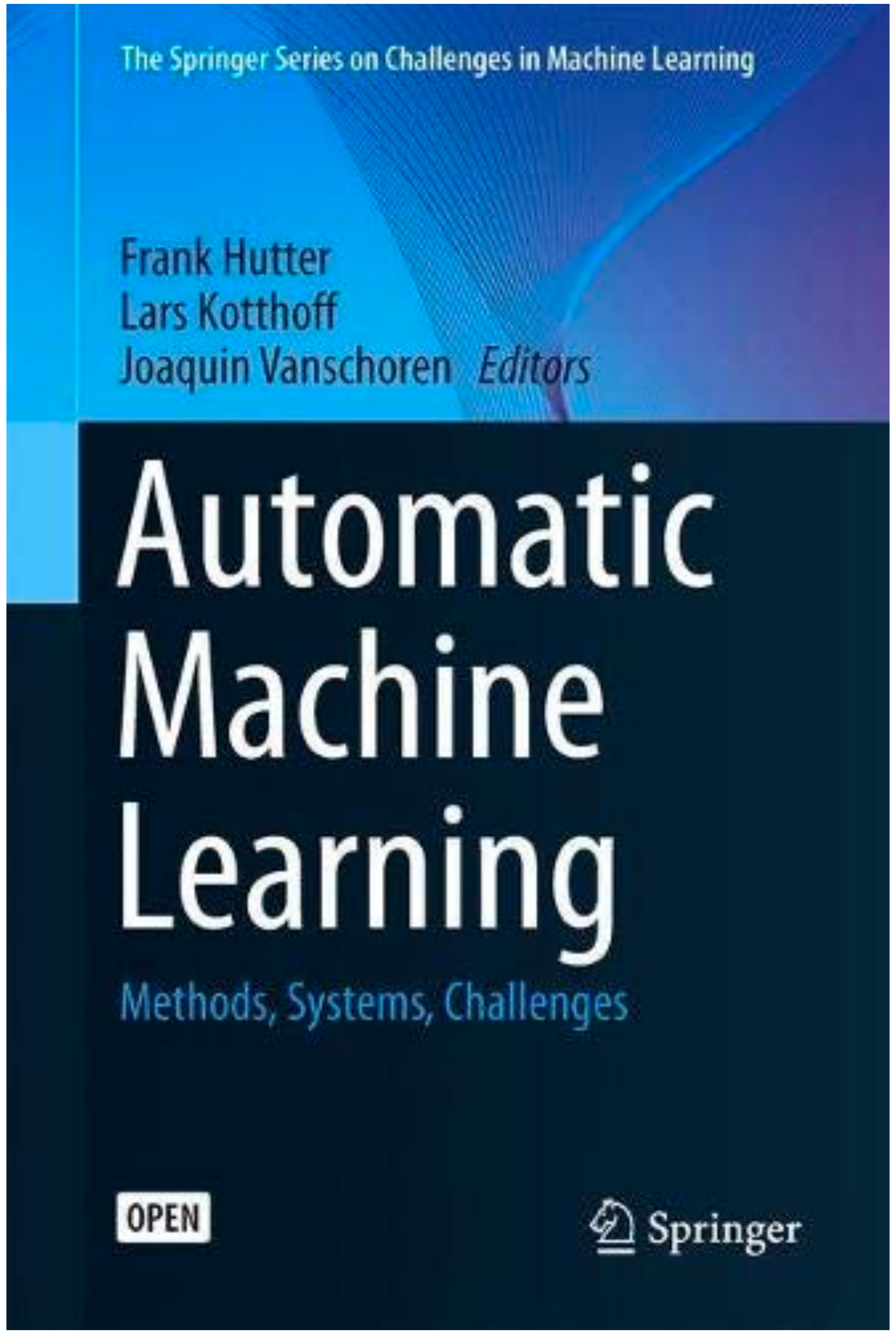
Meta-learning AutoML in practice

- We need a meta-data repository of prior machine learning datasets (tasks) and experiments
  - e.g. OpenML.org
- Ideally, a shared memory that all AutoML tools can access



Further reading

Open access book
PDF (free): www.automl.org/book
www.amazon.de/dp/3030053172



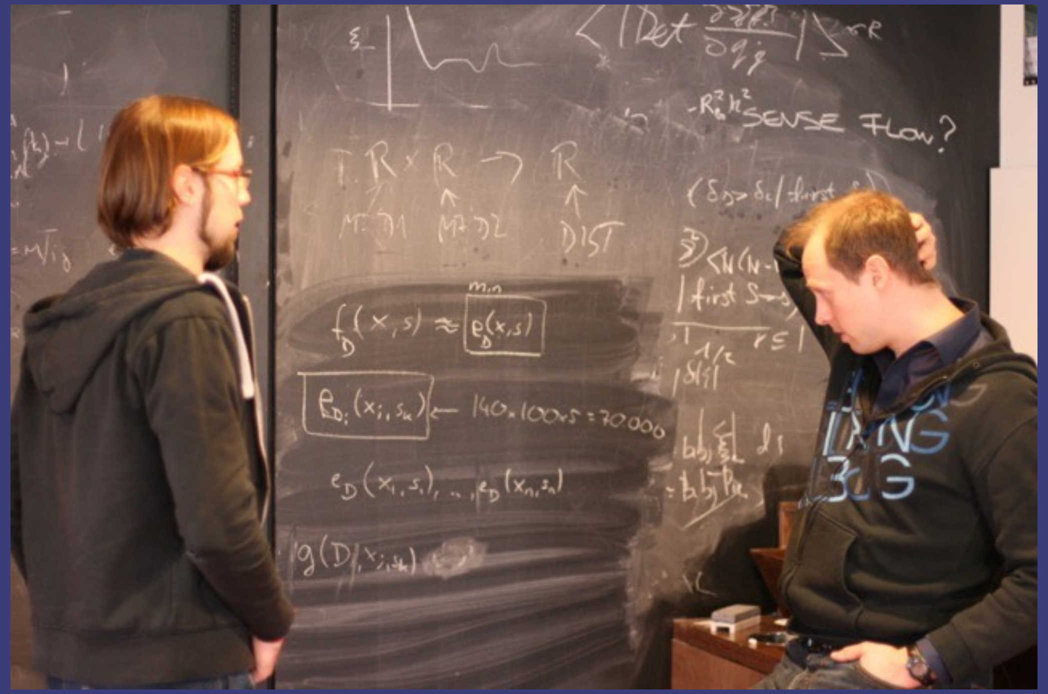
Join us! (and change the world)

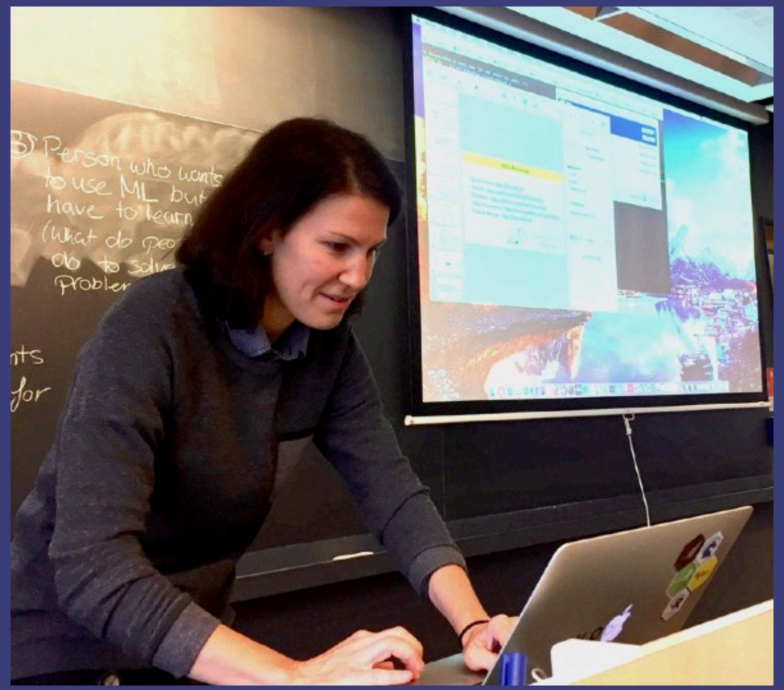
Active open source community
- Hackathons 2-3x a year

OpenML Foundation
- Sponsorship, science

OpenML spin-off: PortML
- Services, projects

PortML





Thanks to the entire OpenML star team


Jan van Rijn


Matthias Feurer


Heidi Seibold


Bernd Bischl


Andreas Müller


Prabhant Singh


Guiseppe Casalicchio


Michel Lang


Pieter Gijsbers


Erin Ledell


Bilge Celik


Janek Thomas


Sahithya Ravi


Marcel Wever


Neil Lawrence


Markus Weimer

and many more!

Thank you!

谢谢

@open_ml

OpenML

www.openml.org