


Automated Machine Learning (AutoML): A Tutorial

Matthias Feurer

University of Freiburg
feurerm@cs.uni-freiburg.de

 @__mfeurer__

Frank Hutter

University of Freiburg & Bosch
fh@cs.uni-freiburg.de

 @FrankRHutter

 @automlfreiburg

Tutorial based on Chapters 1-3 of the book *Automated Machine Learning*
Slides available at automl.org/events/tutorials -> AutoML Tutorial
(all references are clickable links)

Motivation: Successes of Deep Learning

Speech recognition



Computer vision in self-driving cars

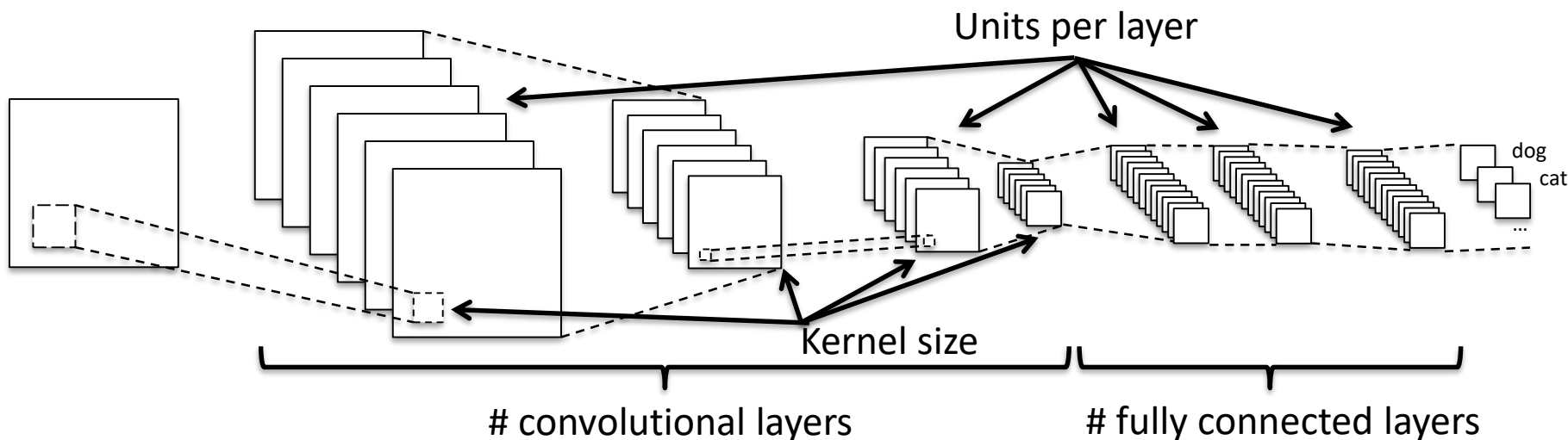


Reasoning in games

One Problem of Deep Learning

Performance is very **sensitive** to **many hyperparameters**

- Architectural hyperparameters

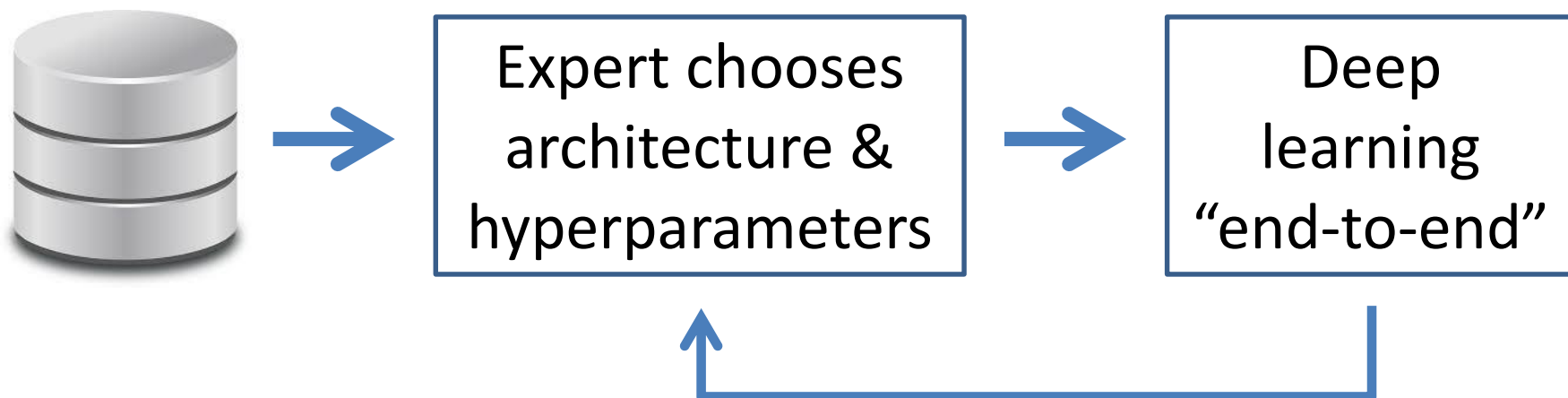


- Optimization algorithm, learning rates, momentum, batch normalization, batch sizes, dropout rates, weight decay, data augmentation, ...

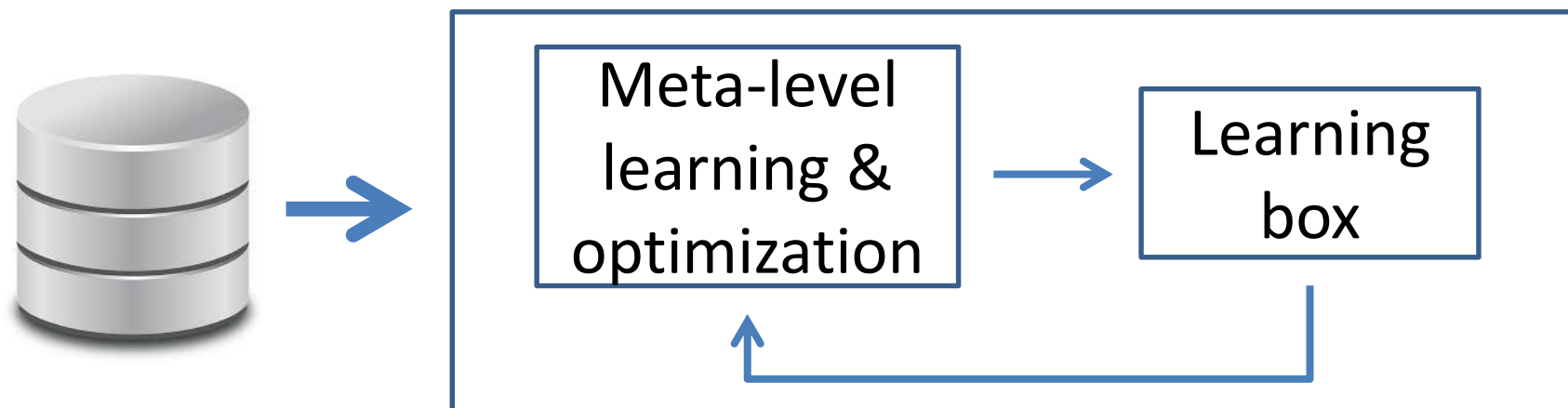
→ **Easily 20-50 design decisions**

Deep Learning and AutoML

Current deep learning practice



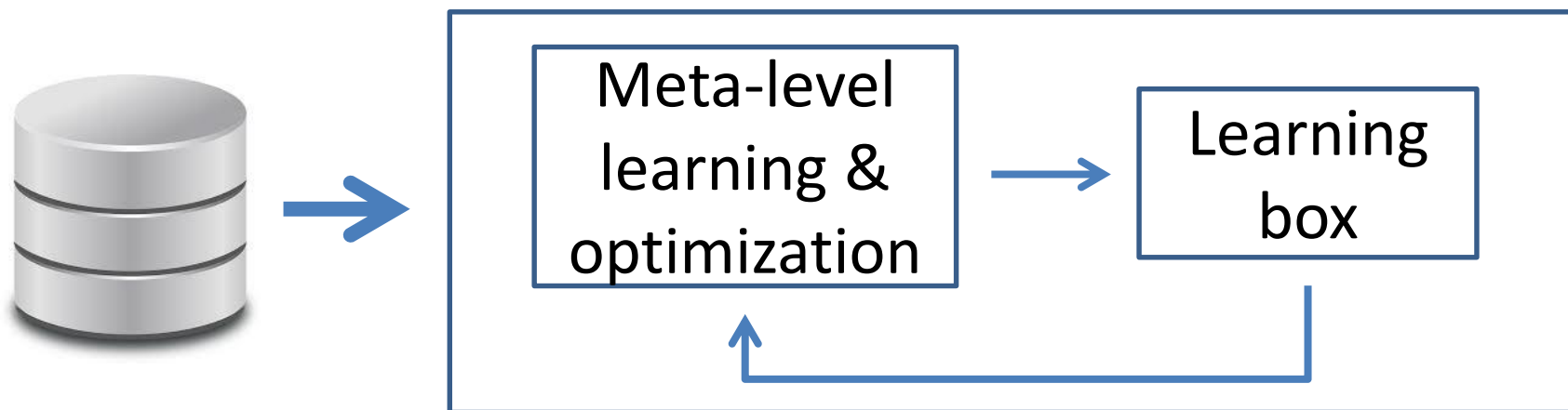
AutoML: true end-to-end learning



Learning box is not restricted to deep learning

- Traditional machine learning pipeline:
 - Clean & preprocess the data
 - Select / engineer better features
 - Select a model family
 - Set the hyperparameters
 - Construct ensembles of models
 - ...

AutoML: true end-to-end learning



Part 1: General AutoML (Matthias, today)

1. AutoML by Hyperparameter Optimization
2. Black-box Hyperparameter Optimization
3. Beyond black-box optimization
4. Examples of AutoML
5. Wrap-up & Conclusion

Part 2: Neural Architecture Search (Frank, tomorrow)

1. Search Spaces
2. Black-box Optimization
3. Beyond Black-box Optimization
4. Best Practices

Part 1: General AutoML

1. AutoML by Hyperparameter Optimization
2. Black-box Hyperparameter Optimization
3. Beyond black-box optimization
4. Examples of AutoML
5. Wrap-up & Conclusion

Part 1: General AutoML

1. AutoML by Hyperparameter Optimization
2. Black-box Hyperparameter Optimization
3. Beyond black-box optimization
4. Examples of AutoML
5. Wrap-up & Conclusion

Hyperparameter Optimization

Definition: Hyperparameter Optimization (HPO)

Let

- λ be the hyperparameters of a ML algorithm A with domain Λ ,
- $\mathcal{L}(A_\lambda, D_{train}, D_{valid})$ denote the loss of A , using hyperparameters λ trained on D_{train} and evaluated on D_{valid} .

The **hyperparameter optimization (HPO)** problem is to find a hyperparameter configuration λ^* that minimizes this loss:

$$\lambda^* \in \arg \min_{\lambda \in \Lambda} \mathcal{L}(A_\lambda, D_{train}, D_{valid})$$

Types of Hyperparameters

- Continuous

Example: learning rate in NNs or GBMs

- Integer

Example: #units, #trees in GBM

- Categorical

- Finite domain, unordered

- Example 1: algo \in {SVM, RF, NN}

- Example 2: activation function \in {ReLU, Leaky ReLU, tanh}

- Example 3: operator \in {conv3x3, separable conv3x3, max pool, ...}

- Special case: binary

Conditional hyperparameters

Conditional hyperparameters B are only active if other hyperparameters A are set a certain way

- Example 1:
 - A = choice of optimizer (Adam or SGD)
 - B = Adam's second momentum hyperparameter (only active if A=Adam)
- Example 2:
 - A = number of layers in a deep neural network
 - B = number of units in layer k (only active if $A \geq k$)
- Example 3:
 - A = choice of classifier (RF or SVM)
 - B = SVM's kernel hyperparameter (only active if A = SVM)

AutoML as Hyperparameter Optimization

Definition: Combined Algorithm Selection and Hyperparameter Optimization (CASH)

Let

- $\mathcal{A} = \{A^{(1)}, \dots, A^{(n)}\}$ be a set of algorithms
- $\Lambda^{(i)}$ denote the hyperparameter space of $A^{(i)}$, for $i = 1, \dots, n$
- $\mathcal{L}(A_{\lambda}^{(i)}, D_{train}, D_{valid})$ denote the loss of $A^{(i)}$, using $\lambda \in \Lambda^{(i)}$ trained on D_{train} and evaluated on D_{valid} .

The Combined Algorithm Selection and Hyperparameter Optimization (CASH) problem is to find a combination of algorithm $A^* = A^{(i)}$ and hyperparameter configuration $\lambda^* \in \Lambda^{(i)}$ that minimizes this loss:

$$A_{\lambda^*}^* \in \arg \min_{A^{(i)} \in \mathcal{A}, \lambda \in \Lambda^{(i)}} \mathcal{L}(A_{\lambda}^{(i)}, D_{train}, D_{valid})$$

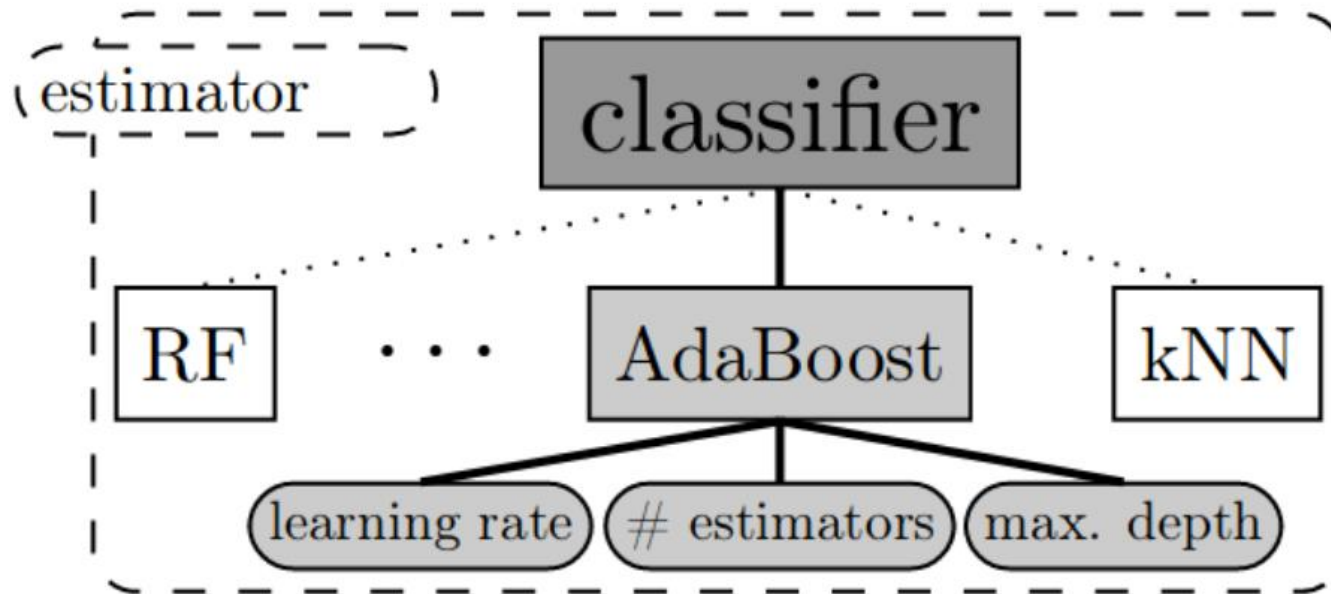
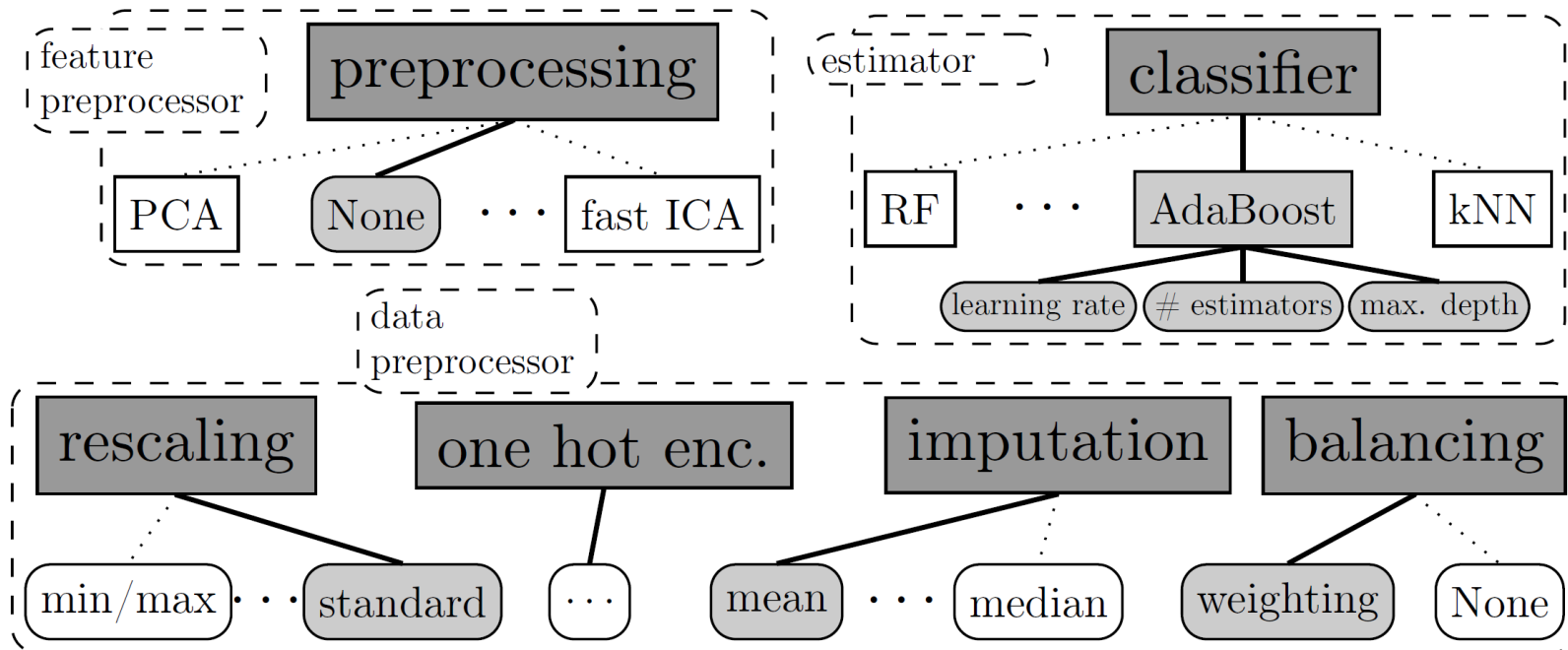


Illustration of the CASH problem in Auto-sklearn:

- 15 base classifiers
- Up to ten hyperparameters each
- Four levels of conditionality

AutoML as Hyperparameter optimization

Not limited to the classification algorithm:

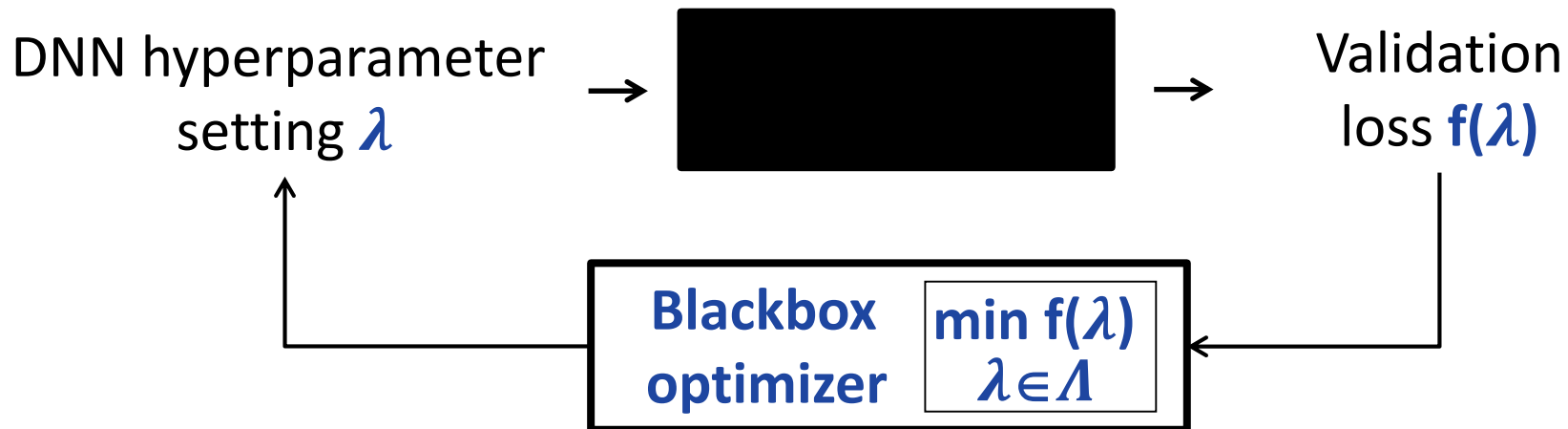


See also [Thornton et al. \(KDD 2013\)](#) which introduced the CASH problem.

Part 1: General AutoML

1. AutoML by Hyperparameter Optimization
2. **Black-box Hyperparameter Optimization**
3. Beyond black-box optimization
4. Examples of AutoML
5. Wrap-up & Conclusion

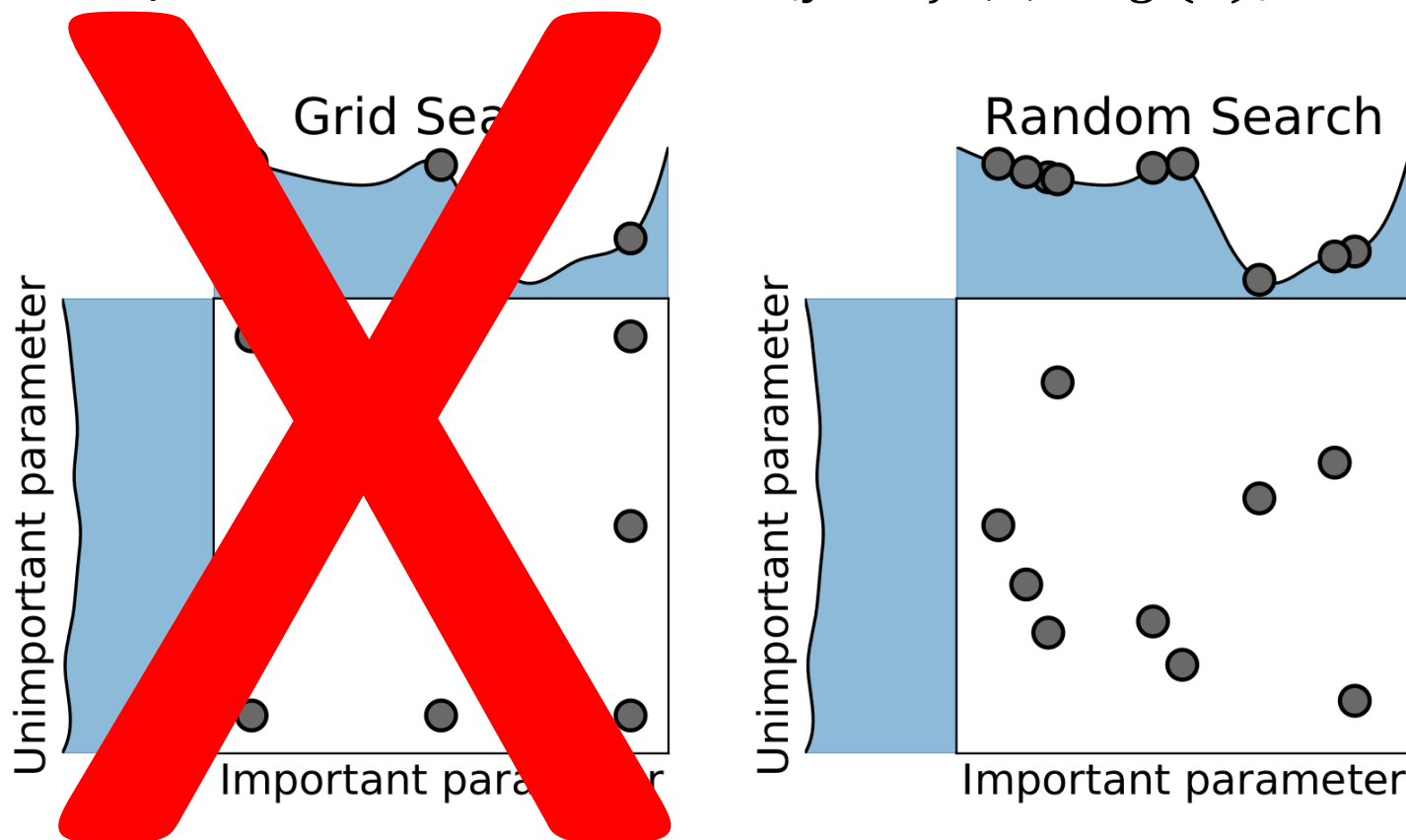
Blackbox Hyperparameter Optimization



- The blackbox function is expensive to evaluate
→ sample efficiency is important

Grid Search and Random Search

- Both completely uninformed
- Grid search suffers from the curse of dimensionality
- Random search handles low intrinsic dimensionality better
- Example: an additive function ($y = f(x) + g(x)$)



Bergstra and Bengio, JMLR 2012; Image source: Feurer & Hutter, CC-BY 4.0

Bayesian Optimization

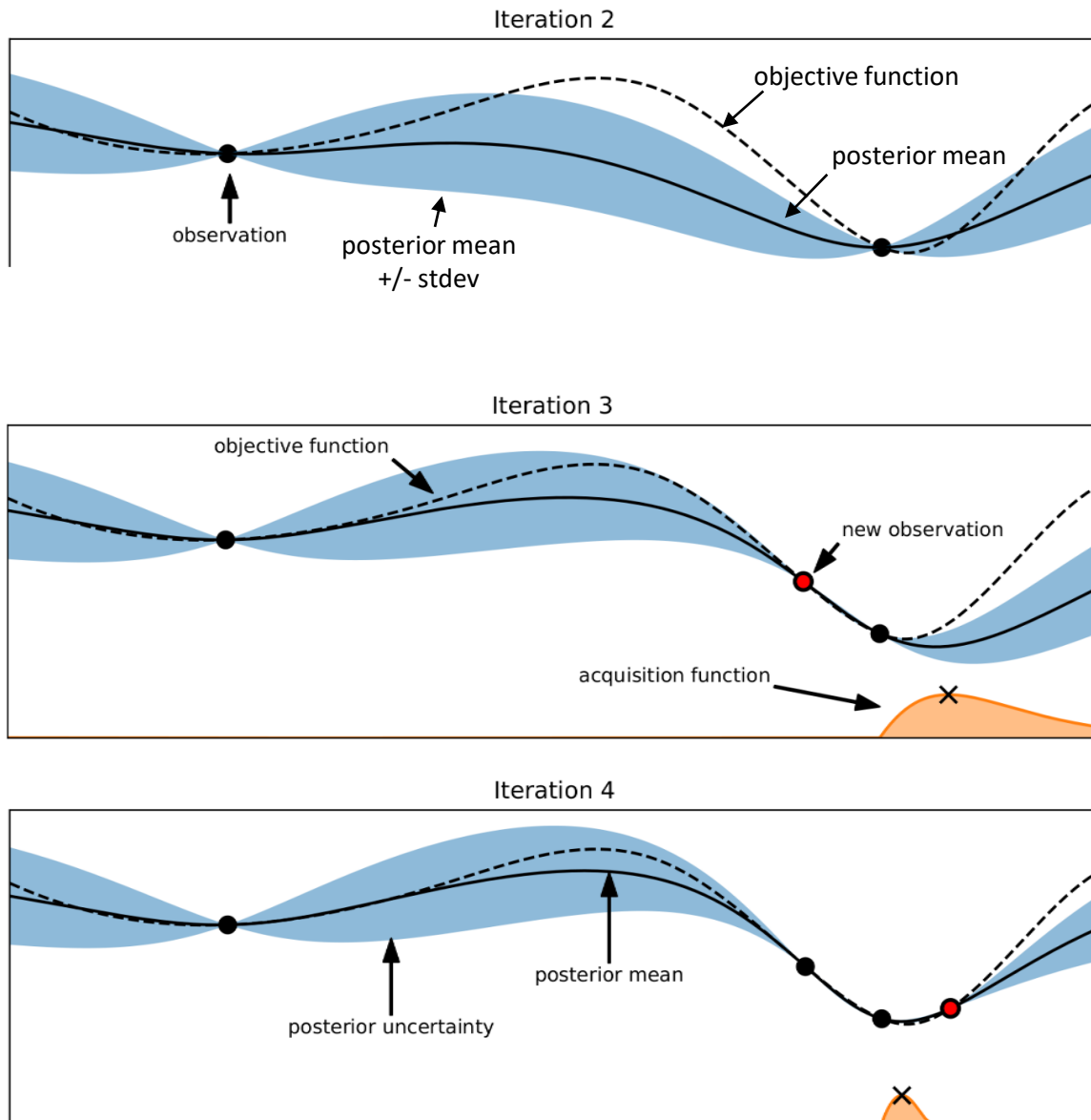
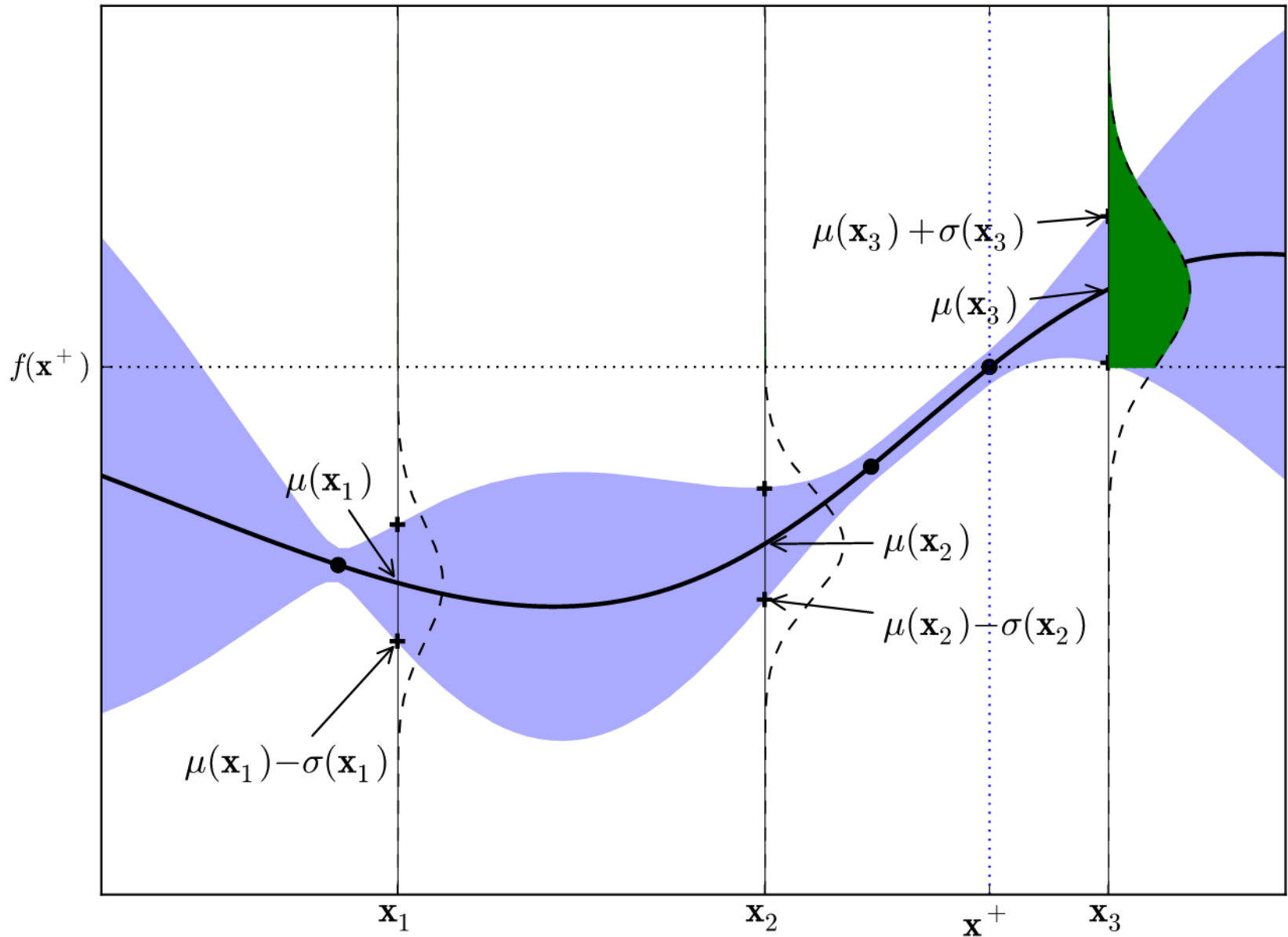


Image source: Feurer & Hutter, CC-BY 4.0

Acquisition Function: Expected Improvement

Image source: Brochu et al., arXiv:1012.2599



Bayesian Optimization

Approach

- Conduct an initial design
- Iteratively:
 - Fit a probabilistic model to the function evaluations $\langle \lambda, f(\lambda) \rangle$, most often a Gaussian process
 - Use that model to trade off Exploration vs. Exploitation in an acquisition function

Popular since Mockus [1974]

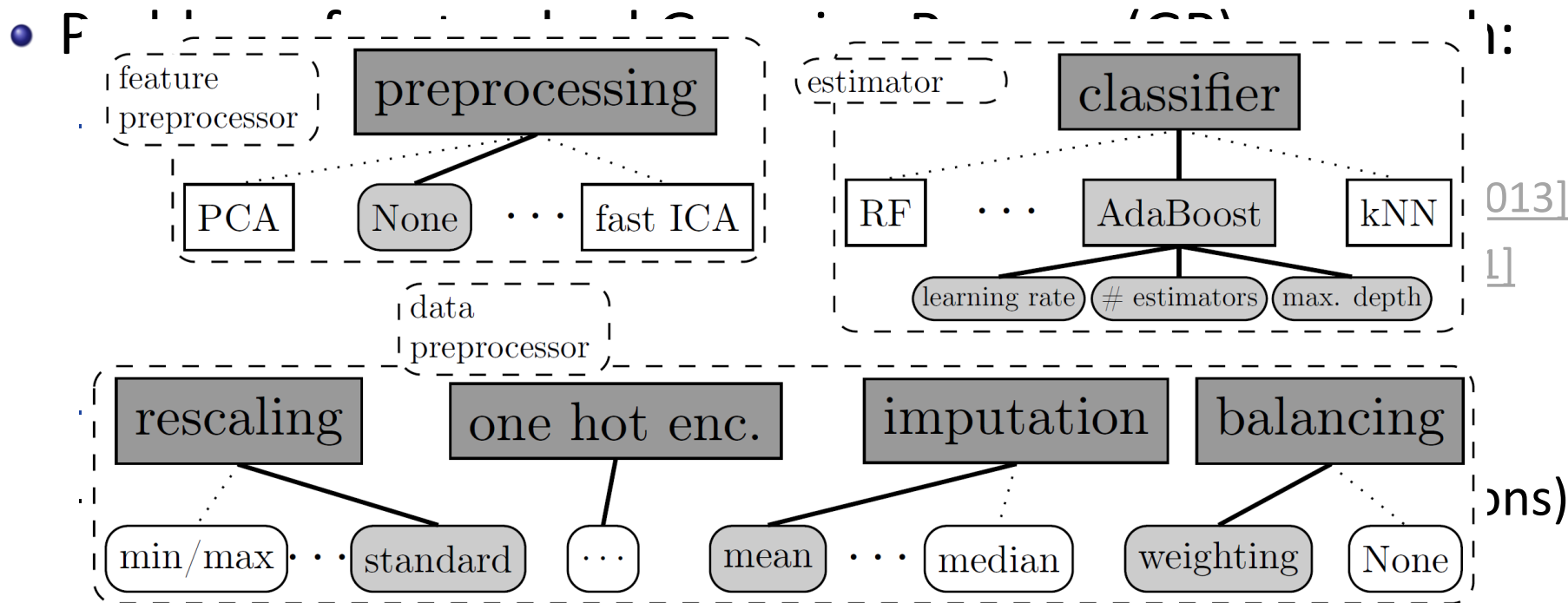
- Sample-efficient
- Works when objective is nonconvex, noisy, has unknown derivatives, etc
- Recent convergence results
[Srinivas et al, 2010; Bull 2011; de Freitas et al, 2012; Kawaguchi et al, 2016; Nguyen et al., 2017; Berkenkamp et al., 2019]
- Excellent reviews by Shahriari et al. (IEEE, 2016) and Frazier (arXiv:1807.02811)

Example: Bayesian Optimization in AlphaGo

- During the development of AlphaGo, its many hyperparameters were tuned with Bayesian optimization multiple times.
- This automatic tuning process resulted in substantial improvements in playing strength. For example, prior to the match with Lee Sedol, we tuned the latest AlphaGo agent and this improved its win-rate from 50% to 66.5% in self-play games. This tuned version was deployed in the final match.
- Of course, since we tuned AlphaGo many times during its development cycle, the compounded contribution was even higher than this percentage.

[Chen et al., arXiv:1812.06855]

AutoML Challenges for Bayesian Optimization



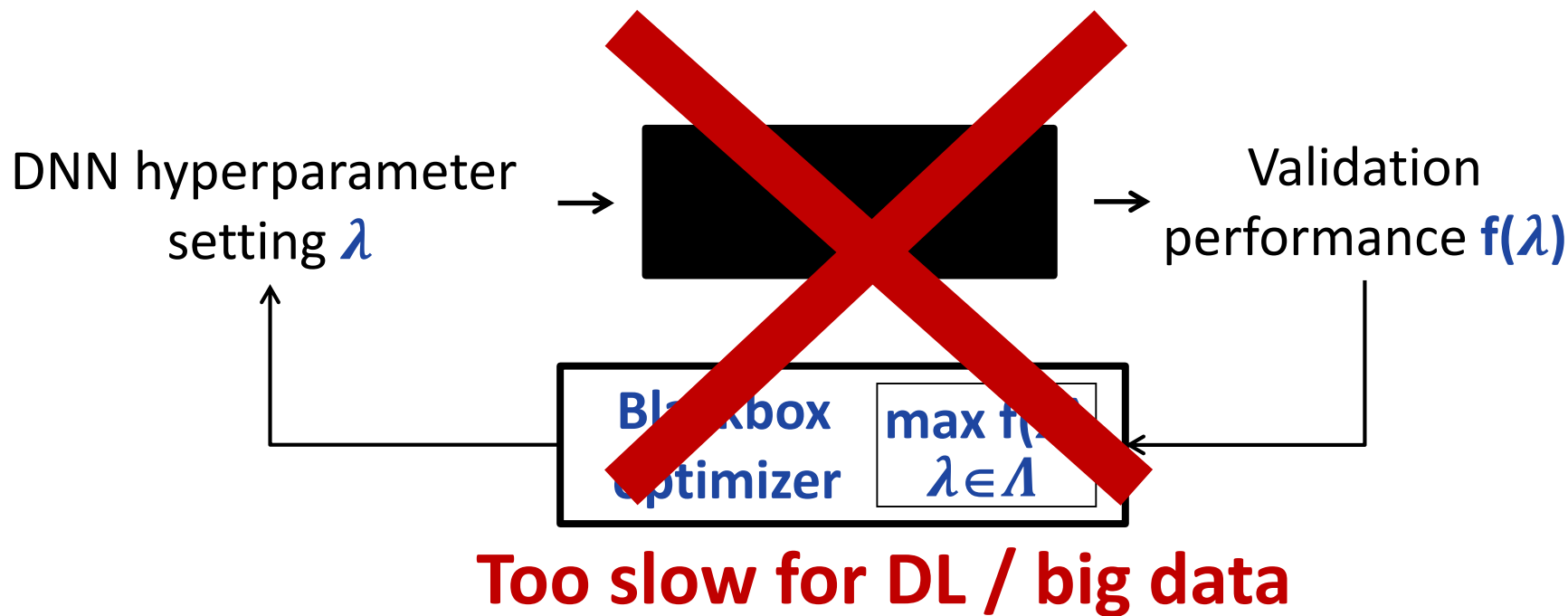
- Simple solution used in SMAC: **random forests** [Breiman, 2001]
 - Frequentist uncertainty estimate: variance across individual trees' predictions [Hutter et al, 2011]

- Two recent promising models for Bayesian optimization
 - Neural networks with **Bayesian linear regression** using the features in the output layer [Snoek et al, ICML 2015]
 - **Fully Bayesian** neural networks, trained with stochastic gradient Hamiltonian Monte Carlo [Springenberg et al, NIPS 2016]
- Tree Parzen Estimator [Bergstra et al., 2011]
 - Non-parametric KDEs for **$p(\lambda \text{ is good})$** and **$p(\lambda \text{ is bad})$** , rather than $p(y|\lambda)$
 - Ratio is proportional to Expected Improvement
- Population-based methods
 - Genetic algorithms, evolutionary algorithms, evolutionary strategies, particle swarm optimization
 - Embarassingly parallel, conceptually simple
- See Chapter 1 of the AutoML book for more information.

Part 1: General AutoML

1. AutoML by Hyperparameter Optimization
2. Black-box Hyperparameter Optimization
3. **Beyond black-box optimization**
4. Examples of AutoML
5. Wrap-up & Conclusion

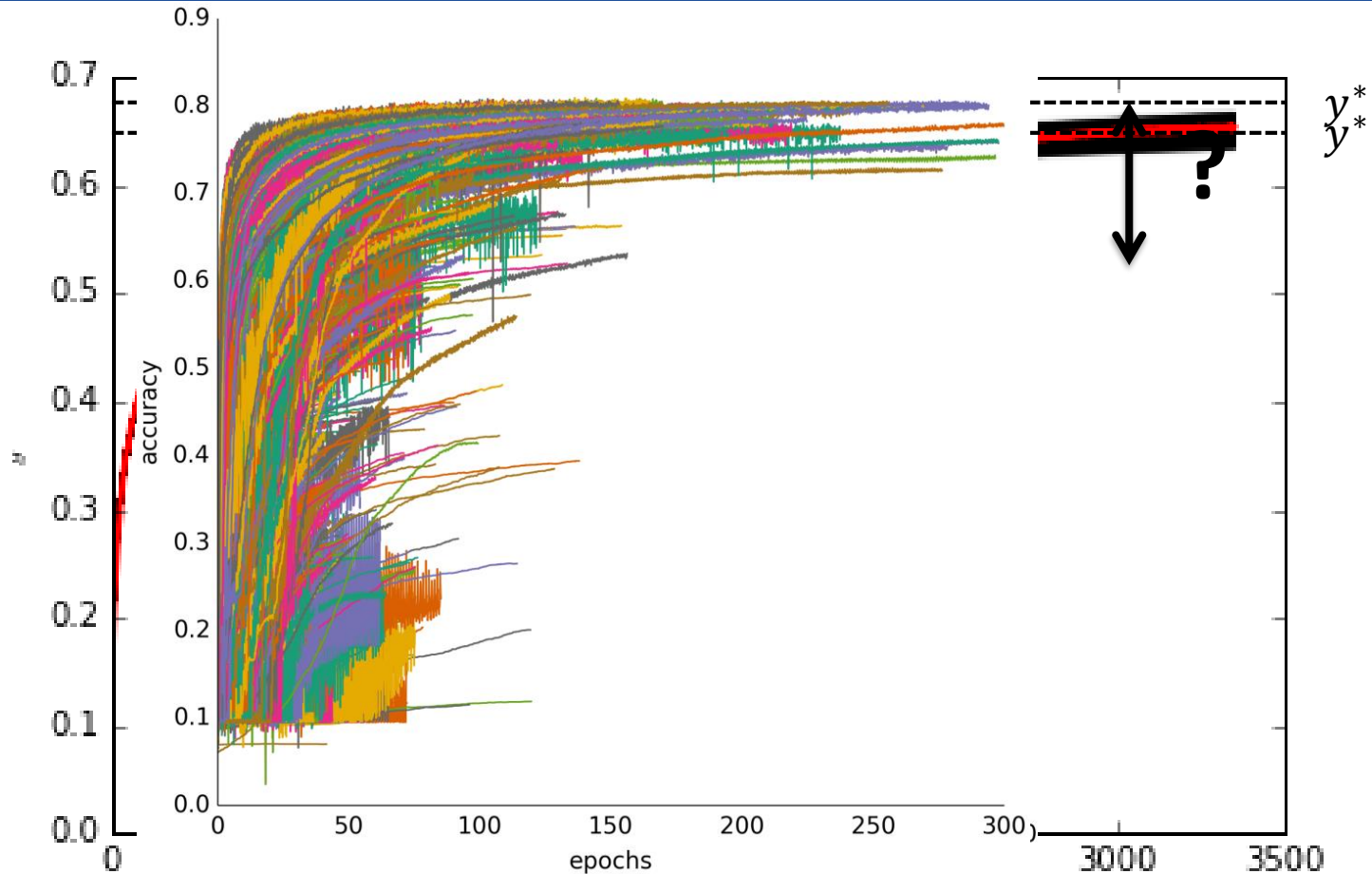
Beyond Blackbox Hyperparameter Optimization



Main Approaches Going Beyond Blackbox HPO

- Extrapolation of learning curves
- Multi-fidelity optimization
- Meta-learning [if there's time left]
- Hyperparameter gradient descent [see AutoML book]

Probabilistic Extrapolation of Learning Curves



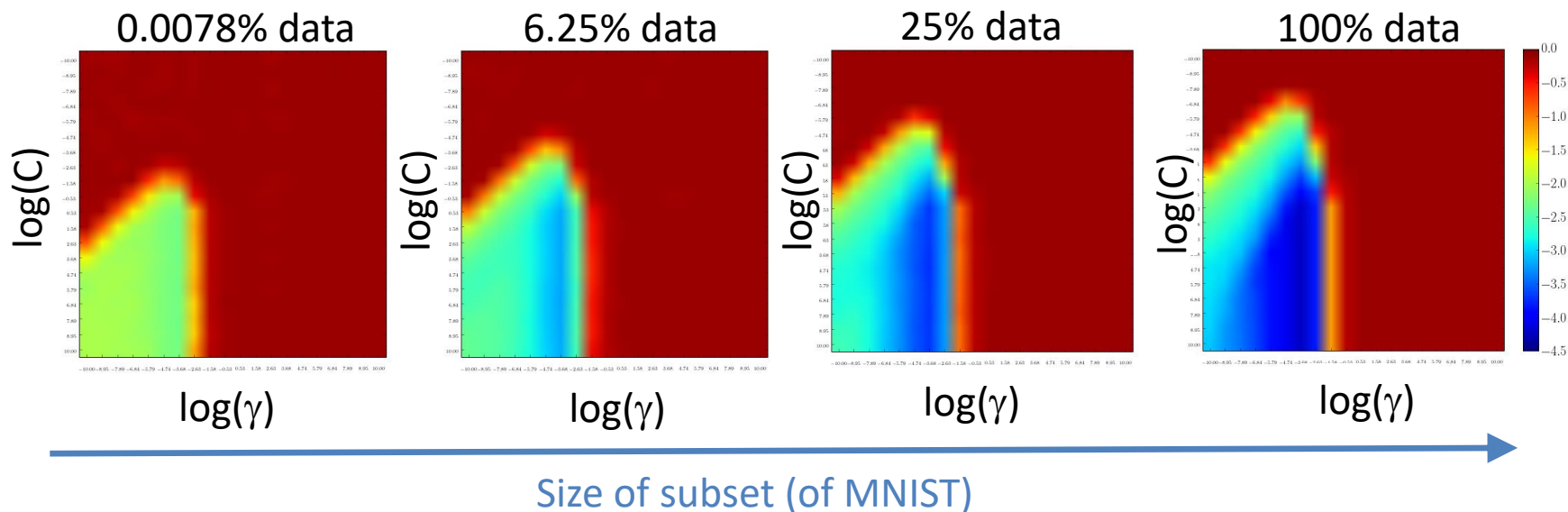
- Parametric learning curve models [\[Domhan et al, IJCAI 2015\]](#)
- Bayesian neural networks [\[Klein et al, ICLR 2017\]](#)
- Linear combination of previous curves [\[Chandrashekar and Lane, ECML2017\]](#)

- Use cheap approximations of the blackbox, performance on which correlates with the blackbox, e.g.
 - Subsets of the data
 - Fewer epochs of iterative training algorithms (e.g., SGD)
 - Fewer trials in deep reinforcement learning
 - Downsampled images in object recognition
- Also applicable in different domains, e.g., **fluid simulations**:
 - Less particles
 - Shorter simulations

Multi-fidelity Optimization

- **Make use of cheap low-fidelity evaluations**

- E.g.: subsets of the data (here: SVM on MNIST)

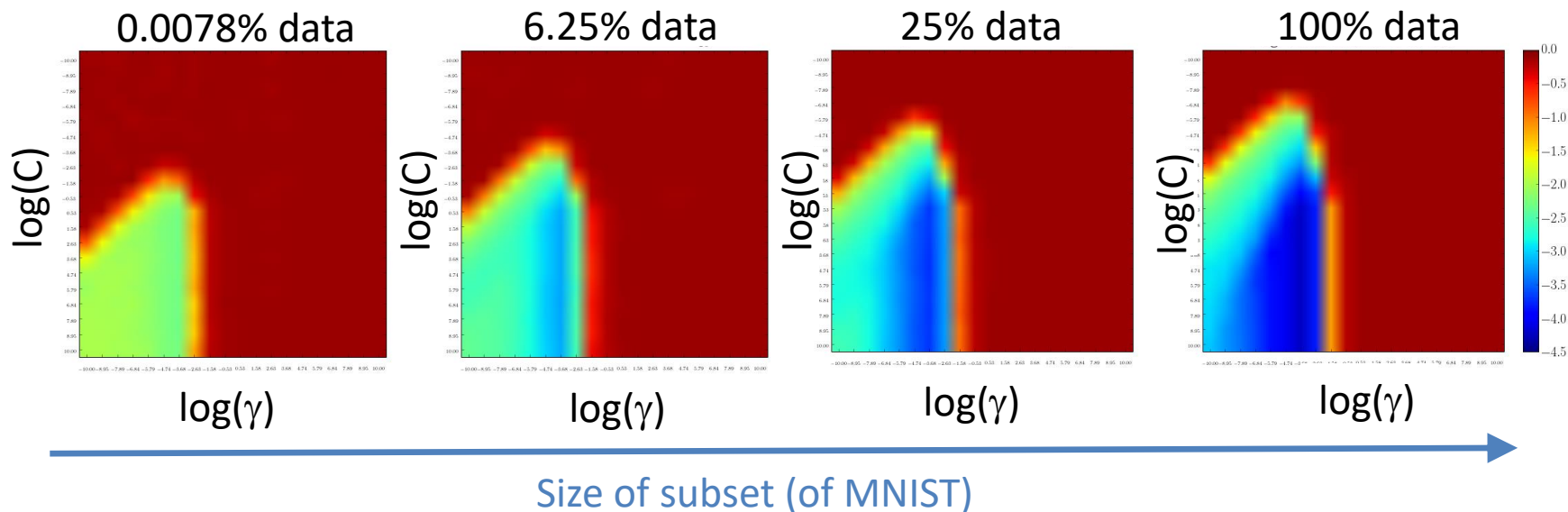


- Many cheap evaluations on small subsets
 - Few expensive evaluations on the full data
 - **Up to 1000x speedups** [Klein et al, AISTATS 2017]

Multi-fidelity Optimization

- **Make use of cheap low-fidelity evaluations**

- E.g.: subsets of the data (here: SVM on MNIST)

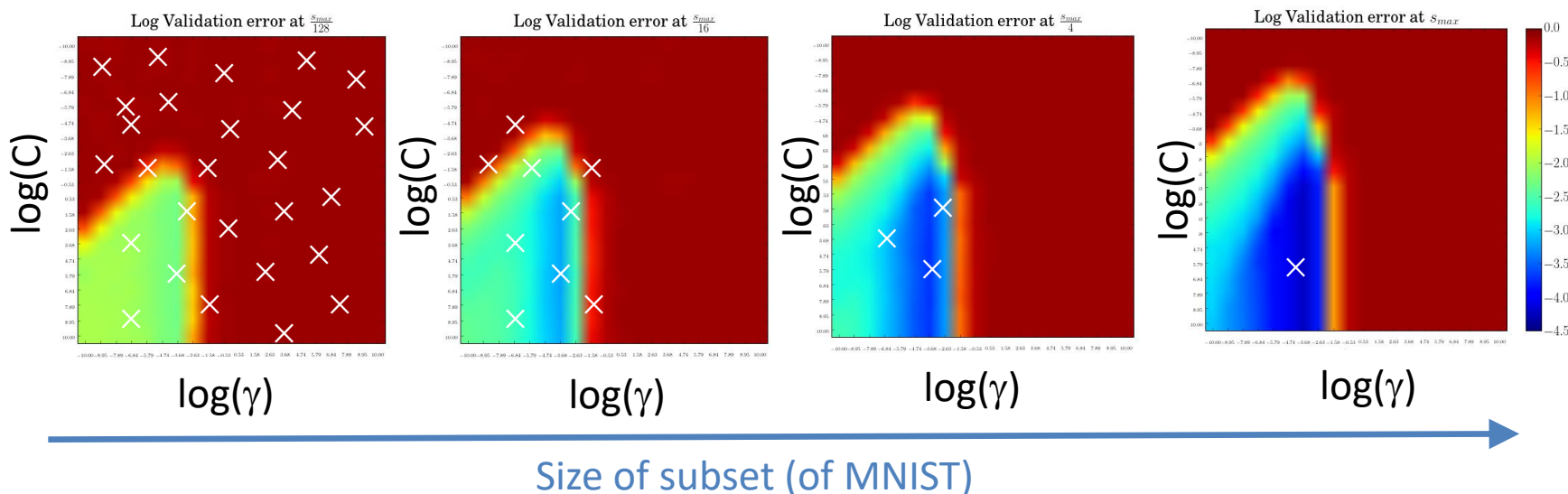


- Fit a Gaussian process model $f(\lambda, b)$ to predict performance as a function of hyperparameters λ and budget b
- Choose both λ and budget b to maximize “bang for the buck”

[Swersky et al, NeurIPS 2013; Swersky et al, arXiv 2014; Klein et al, AISTATS 2017; Kandasamy et al, ICML 2017]

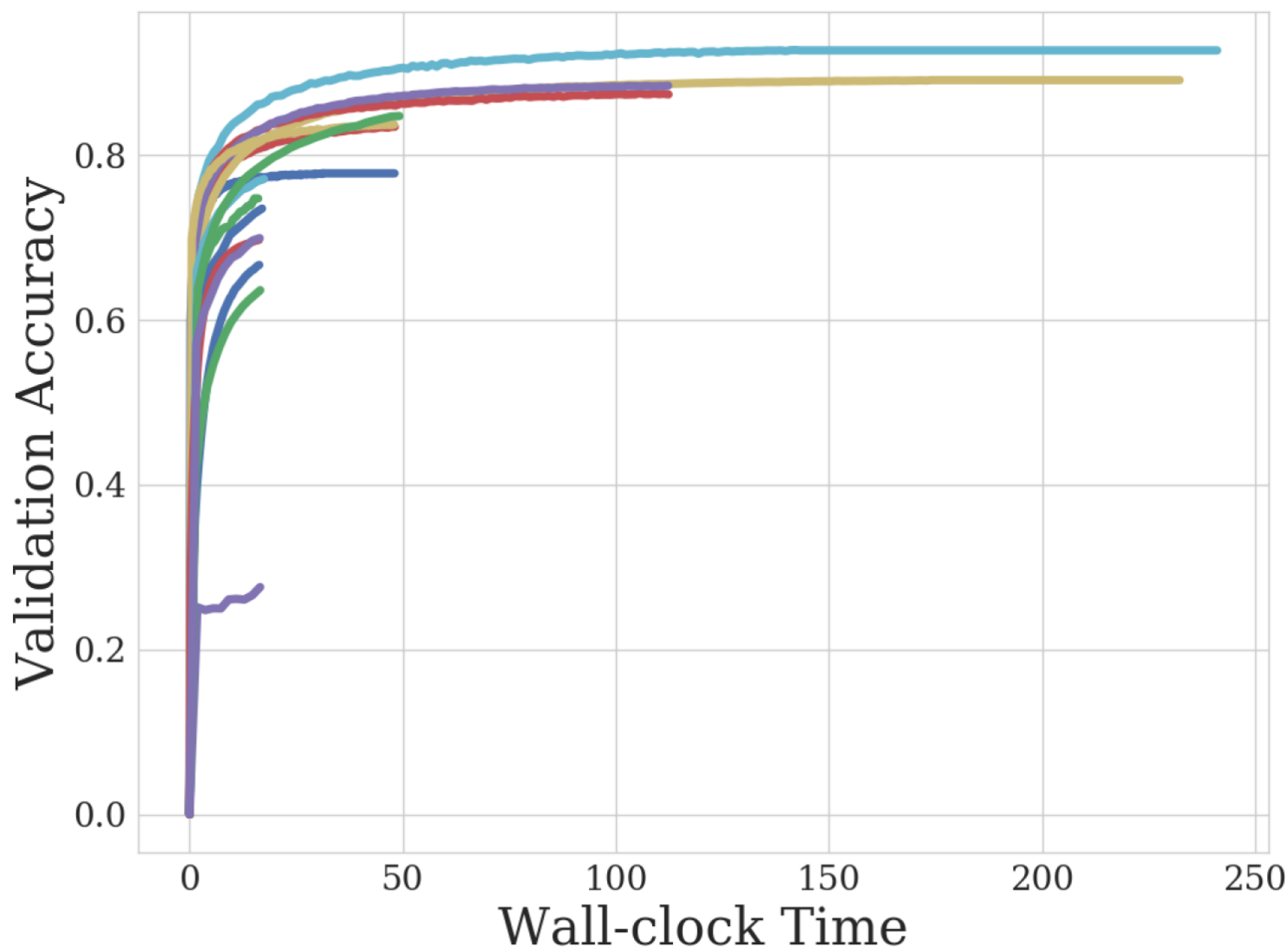
A Simpler Approach: Successive Halving (SH)

- **Idea:** Use a bandit to allocate more budget to promising configurations
- **Successive Halving** [\[Jamieson & Talwalkar, AISTATS 2016\]](#)
 - Randomly sample N configurations & evaluate on cheapest fidelity
 - Keep the top half, double its budget (or top third, triple budget)



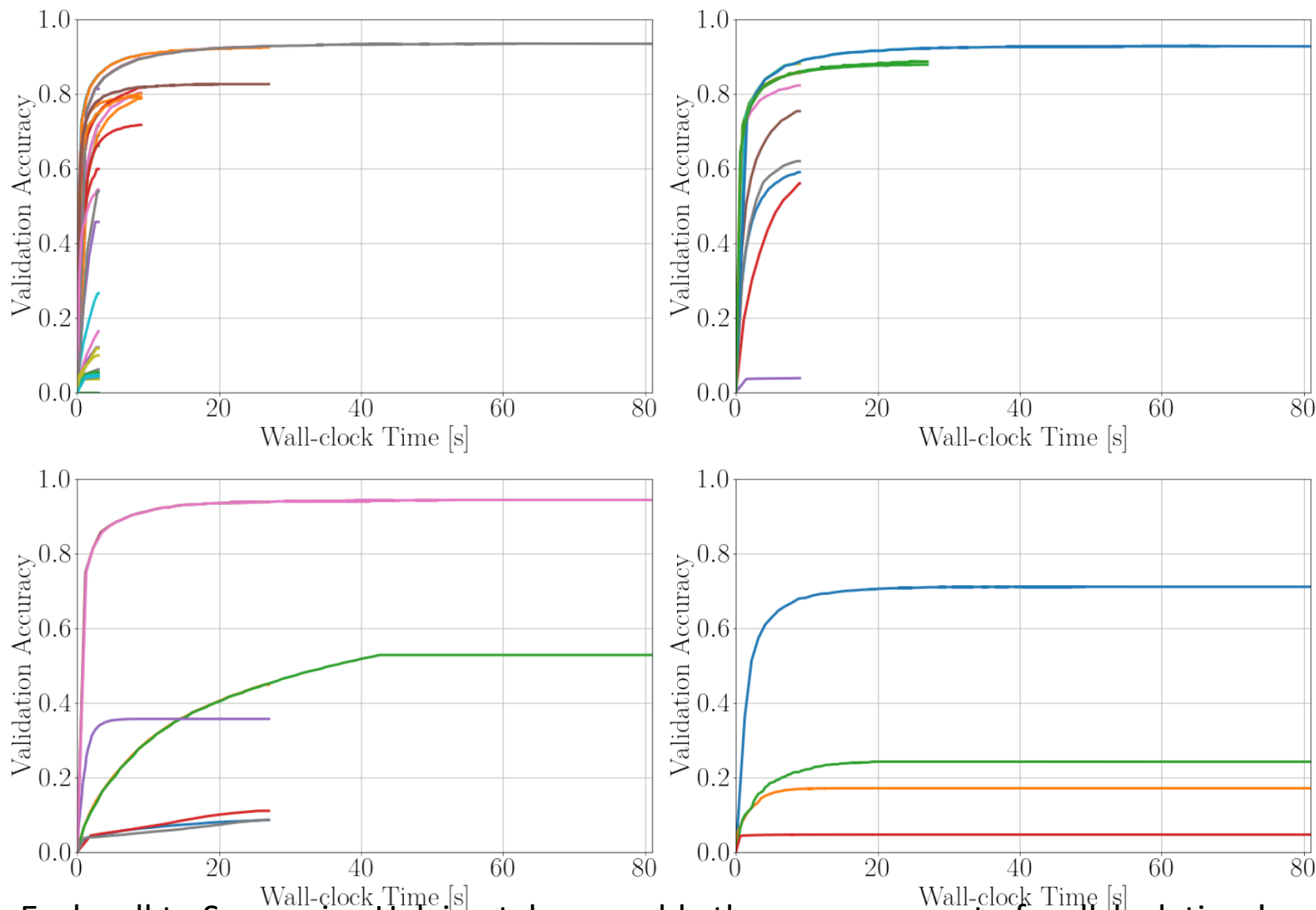
A Simpler Approach: Successive Halving (SH)

[Jamieson & Talwalkar, AISTATS 2016]



Hyperband (its first 4 calls to SH)

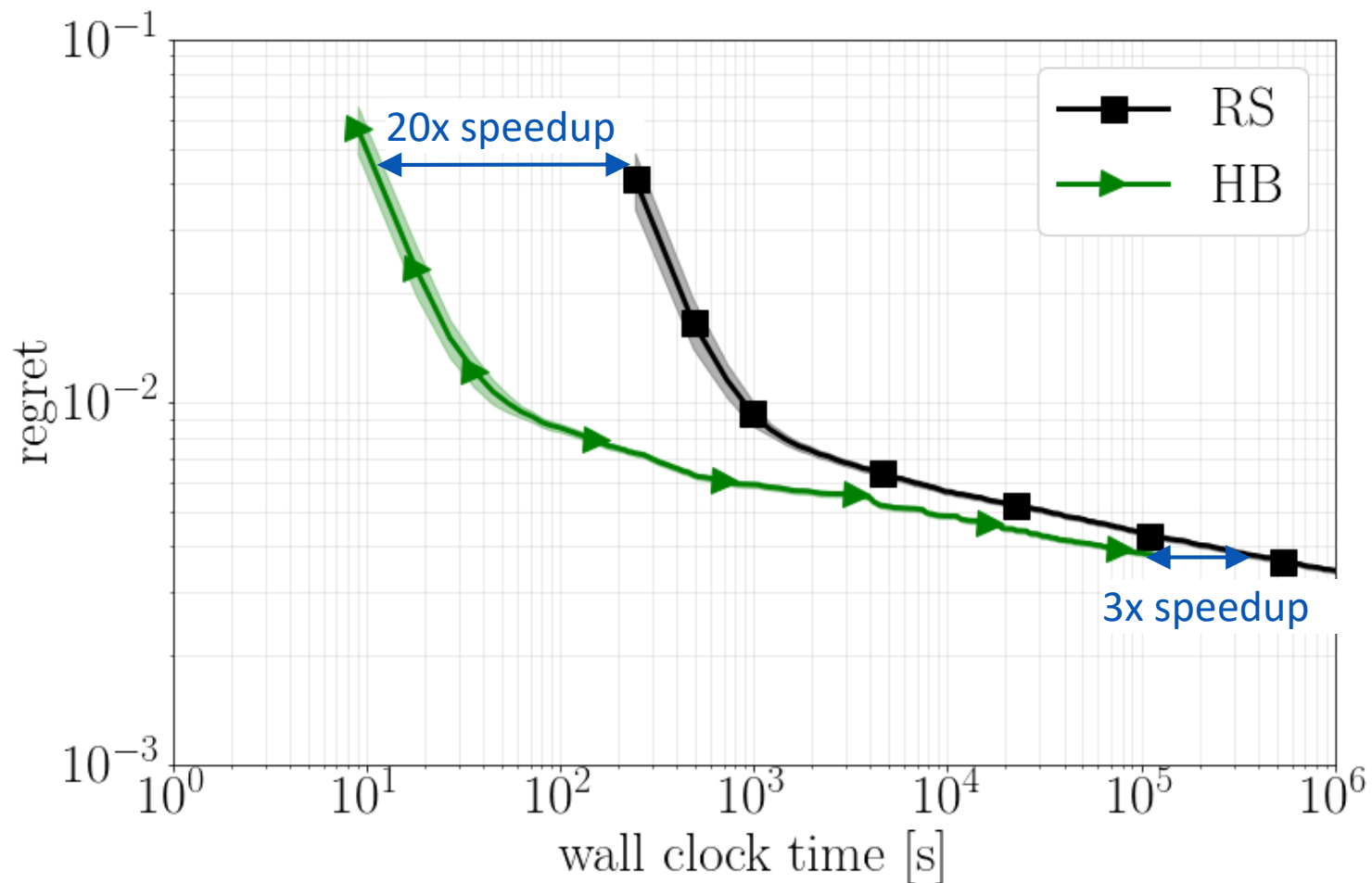
[Li et al, JMLR 2018]



Each call to Successive Halving takes roughly the same amount of wallclock time!

- Advantages of Hyperband
 - Strong anytime performance
 - General-purpose
 - Low-dimensional continuous spaces
 - High-dimensional spaces with conditionality, categorical dimensions, etc
 - Easy to implement
 - Scalable
 - Easily parallelizable
- Advantage of Bayesian optimization: strong final performance
- Combining the best of both worlds in BOHB
 - Bayesian optimization
 - for choosing the configurations to evaluate (using a TPE variant)
 - Hyperband
 - for deciding how to allocate budgets

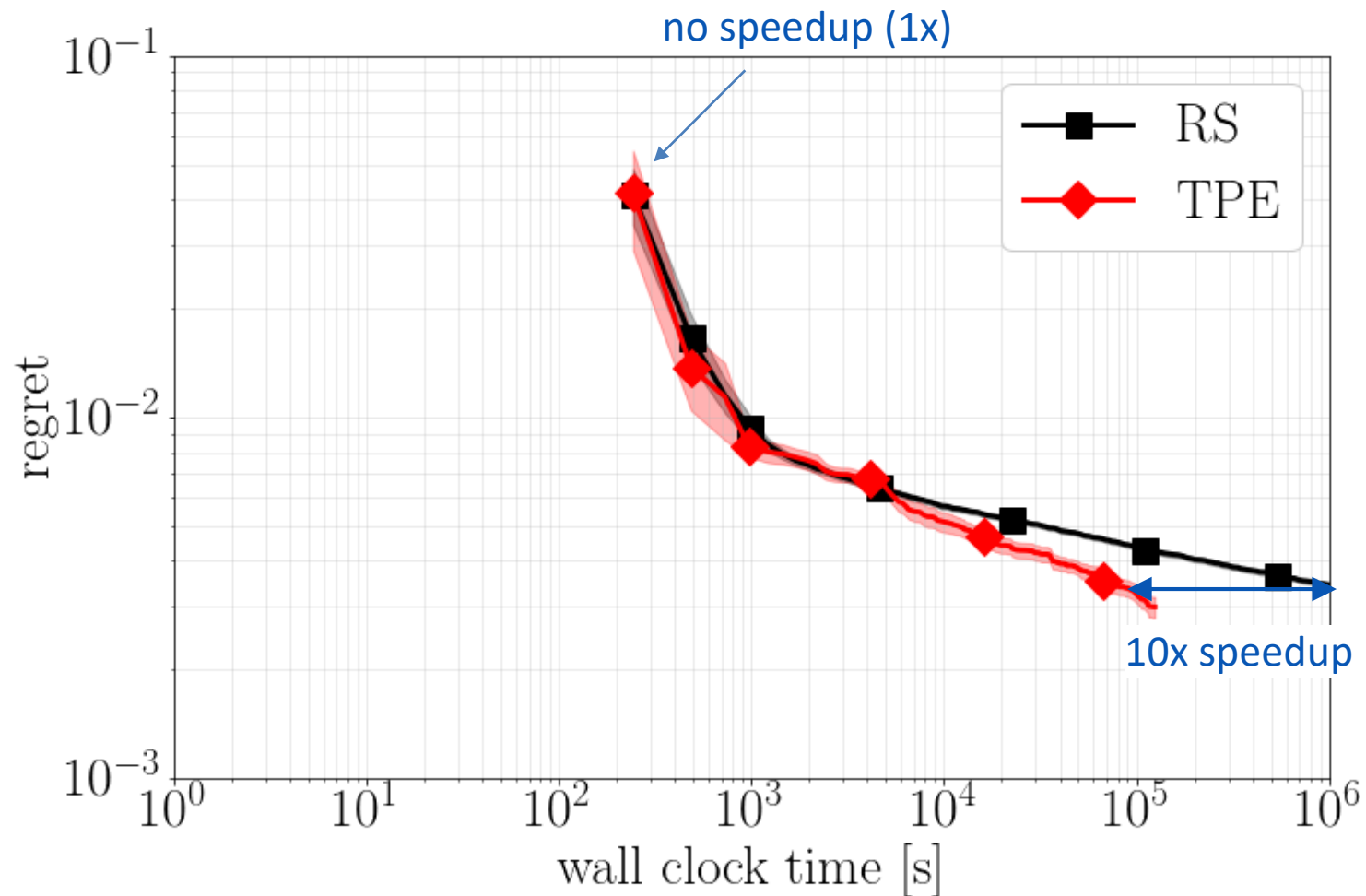
Hyperband vs. Random Search



Biggest advantage: much improved **anytime performance**

Auto-Net on dataset adult

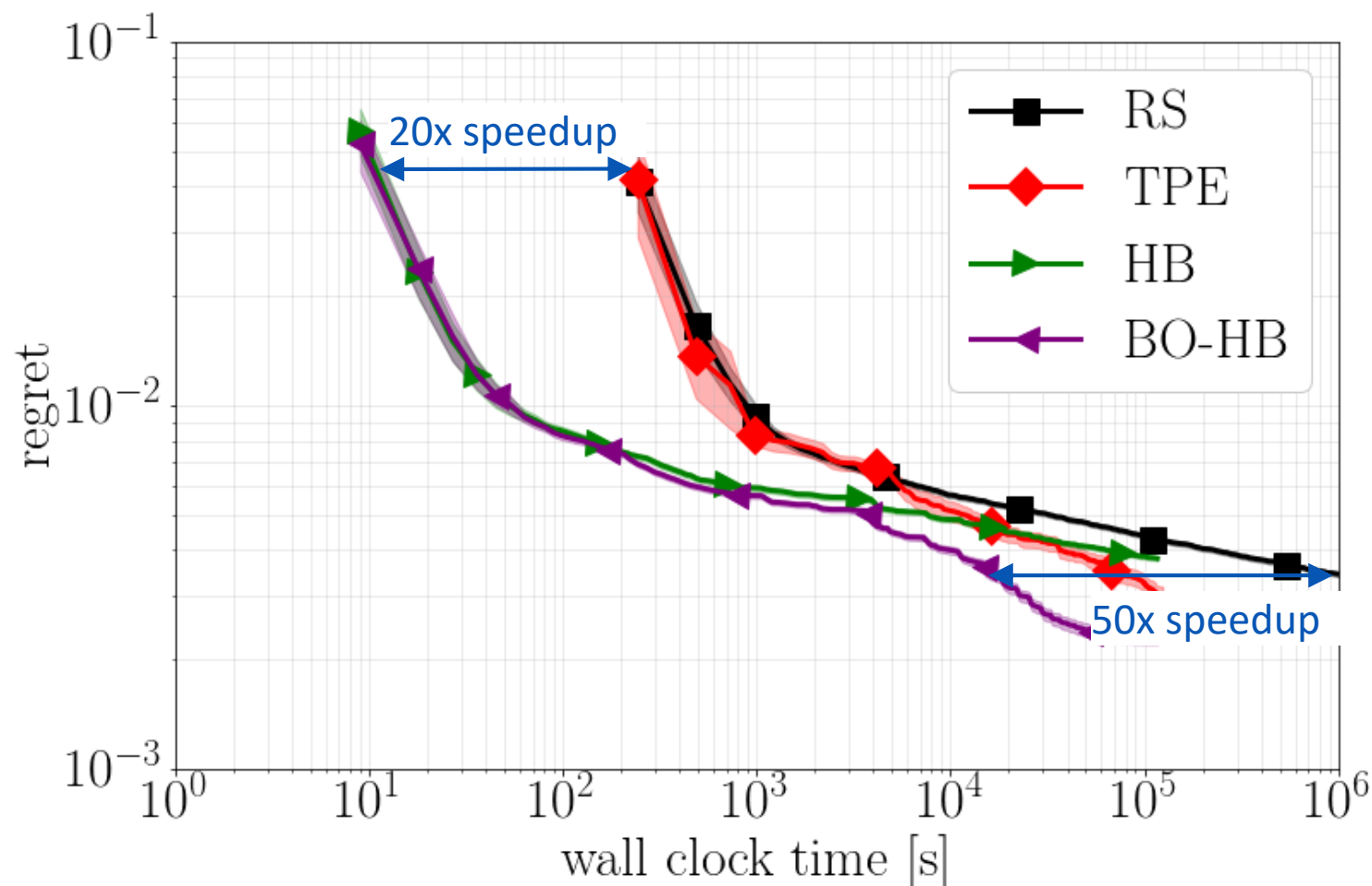
Bayesian Optimization vs Random Search



Biggest advantage: much improved **final performance**

Auto-Net on dataset adult

Combining Bayesian Optimization & Hyperband



Best of both worlds: strong **anytime** and **final performance**

Auto-Net on dataset adult

Part 1: General AutoML

1. AutoML by Hyperparameter Optimization
2. Black-box Hyperparameter Optimization
3. Beyond black-box optimization
4. Examples of AutoML
5. Wrap-up & Conclusion

What can be automated?

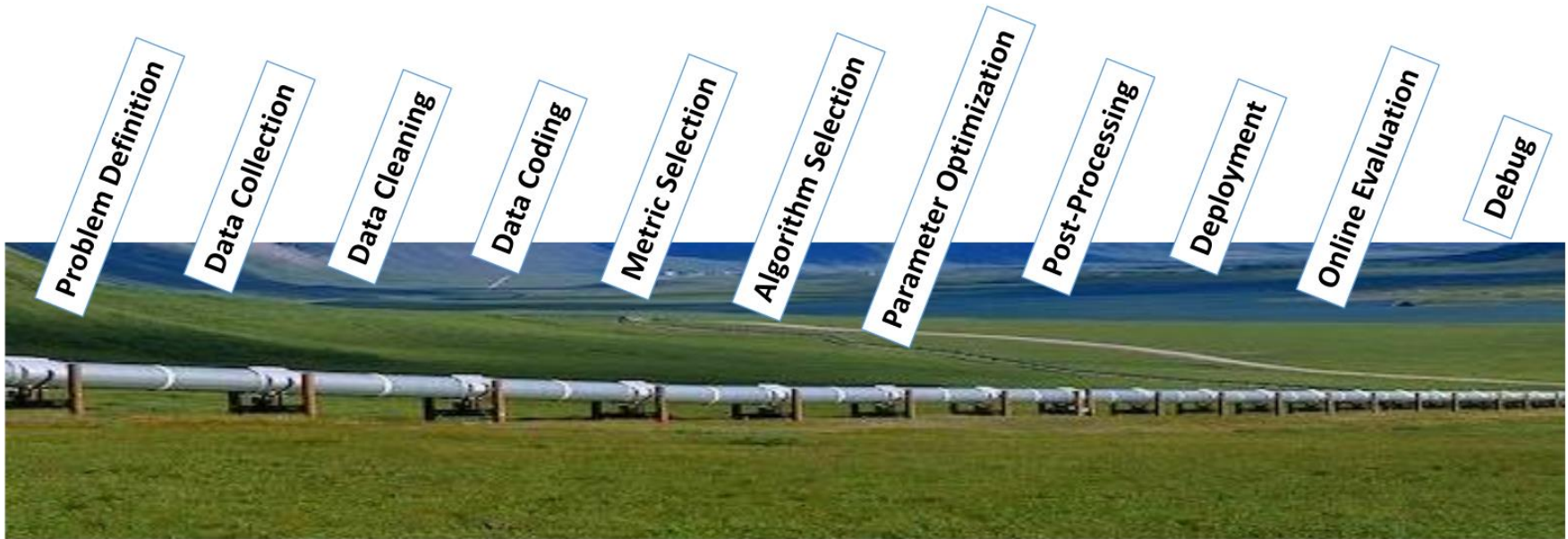


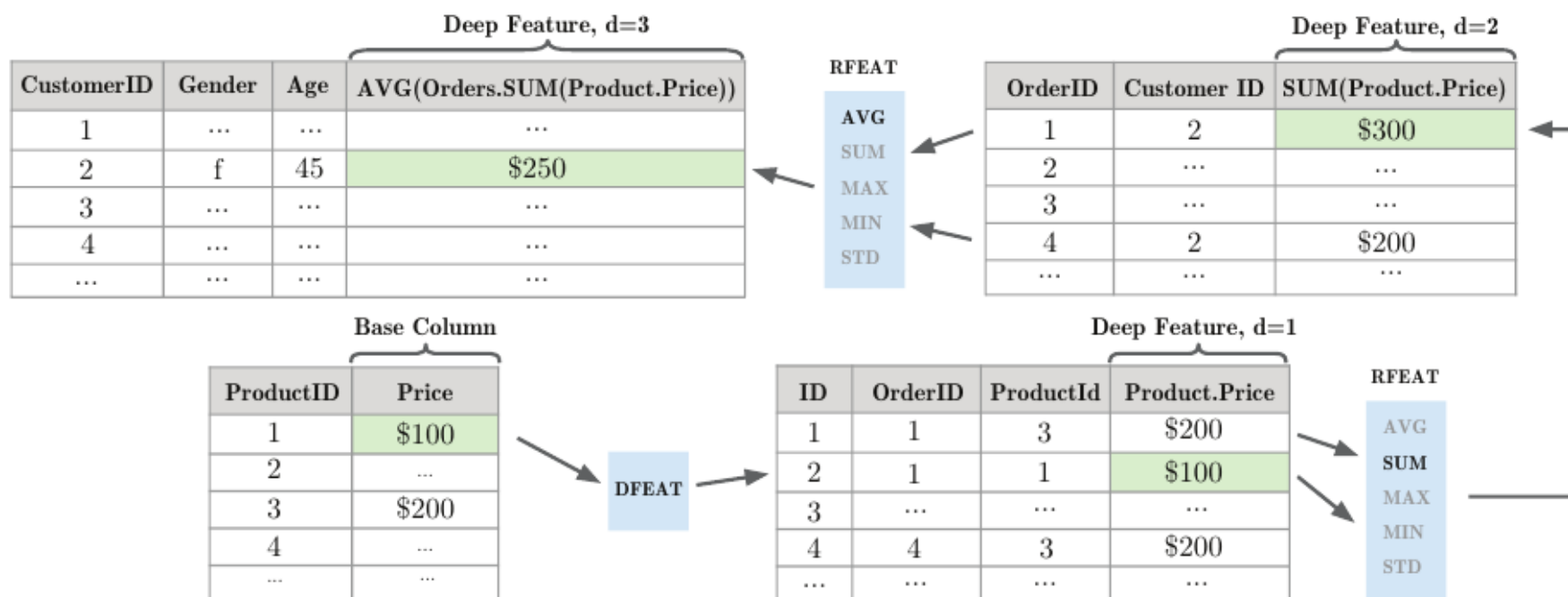
Image credit: Rich Caruana, AutoML 2015

Example I – Data cleaning and ingestion

- Automatically detect the dialect of CSV files
[van den Burg et al., arXiv:1811.11242]
- Automatically classify data types
[Valera and Ghahramani, ICML 2017]
- Automatically detect mistakes in the data gathering process
[Sutton et al., KDD 2018]
- Check out the talk of Charles Sutton@AutoML Workshop 2019

Example II – Feature Engineering

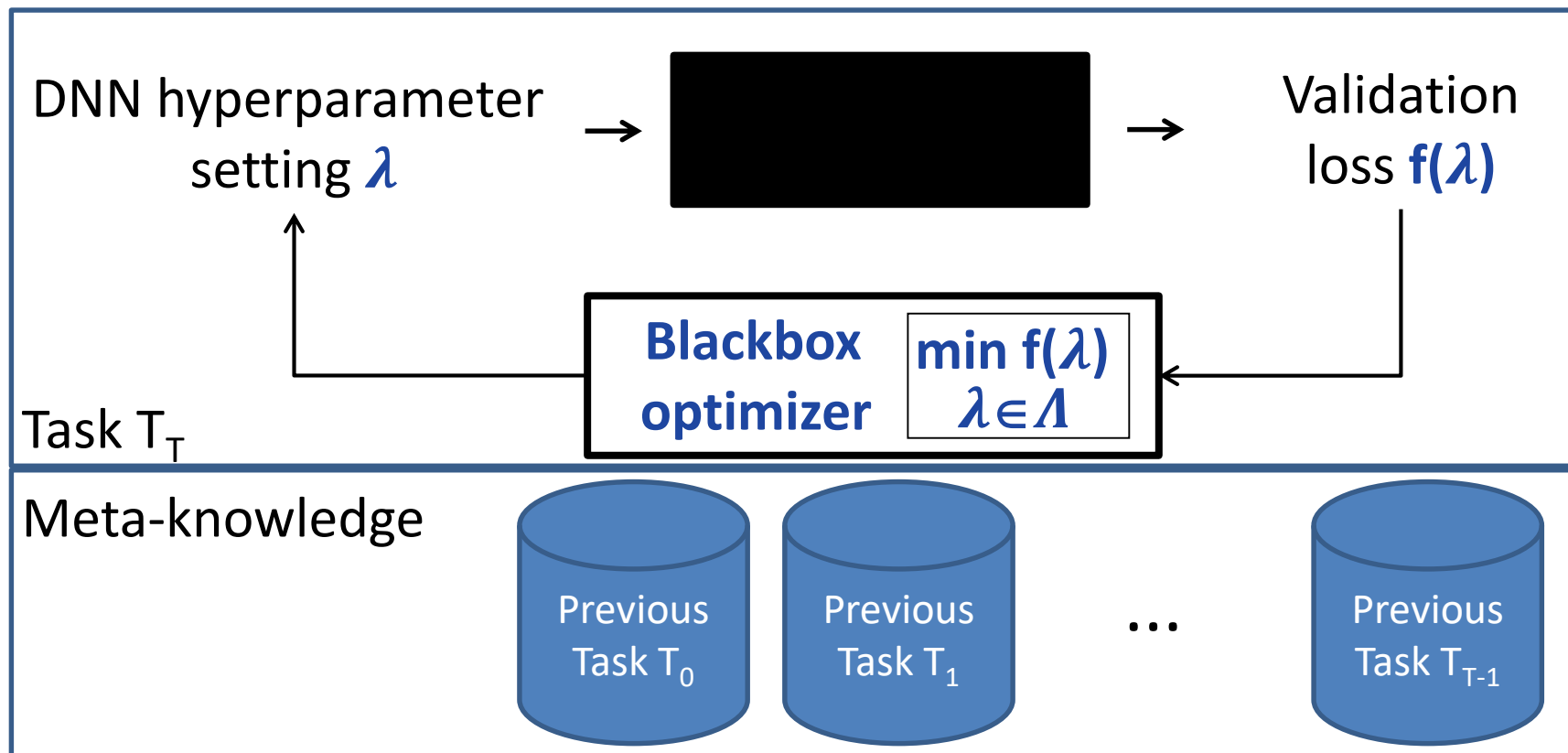
- From relational data bases:
 - Automatically aggregates information, can for example generate the *average sum of orders*
 - Requires post-hoc pruning of the features
 - [Kanter and Veeramachaneni, DSAA 2015]



Example III: Off-the-shelf Algorithms

- Reduce the amount of tuning:
 - Random Forests are excellent default classifiers
 - Learning rate adaption
 - rProp
 - RMSProp
 - ...
 - Adam
 - ...
 - Ranger (look ahead + rectified Adam)
 - Pre-trained Neural Networks
 - Better defaults
 - ...

Example IV: Classical Meta-Learning



- Learning Defaults
- Joint Models for Bayesian Optimization
- Post-hoc analysis of hyperparameter importance

(see slides in the end for further info and links)

Example V: Learning Weight Inizializations

Model-Agnostic Meta-Learning (MAML) [Finn et al., ICML 2017]

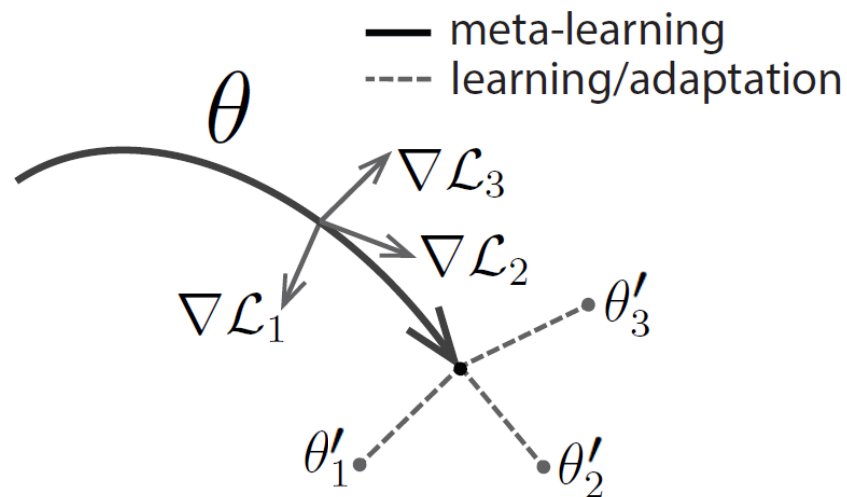
while not done:

1. sample tasks T_i
2. update task weights

$$\theta'_i = \theta - \alpha \nabla_{\theta} \mathcal{L}_{T_i}(f_{\theta})$$

3. update meta weights θ by solving

$$\min_{\theta} \sum_{T_i} \mathcal{L}_{T_i}(f_{\theta'_i}) = \sum_{T_i} \mathcal{L}_{T_i}(f_{\theta - \alpha \nabla_{\theta} \mathcal{L}_{T_i}(f_{\theta})})$$



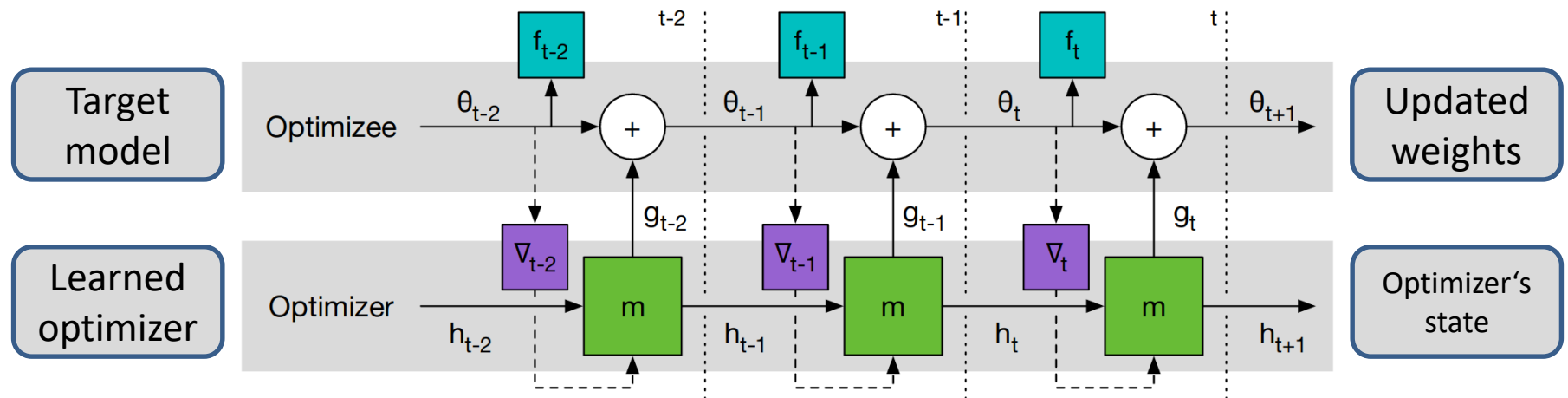
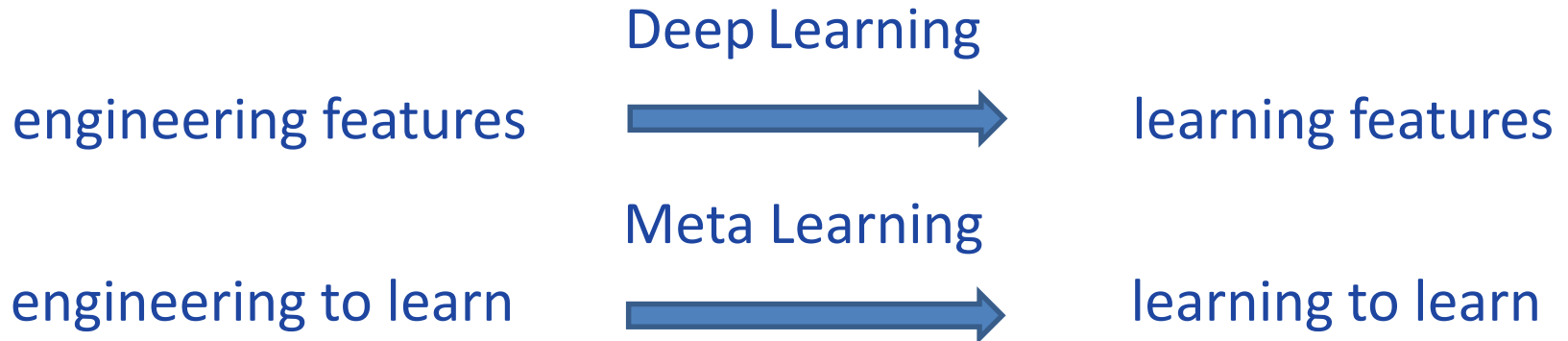
See also

REPTILE [Nichol et al., arXiv preprint 2018]

LEAP [Flennerhag et al., ICLR 2019]

iMAML [Rajeswaran et al., NeurIPS 2019]

Example VI: Learning to Learn



[Andrychowicz et al., NeurIPS 2016]

What can be automated?

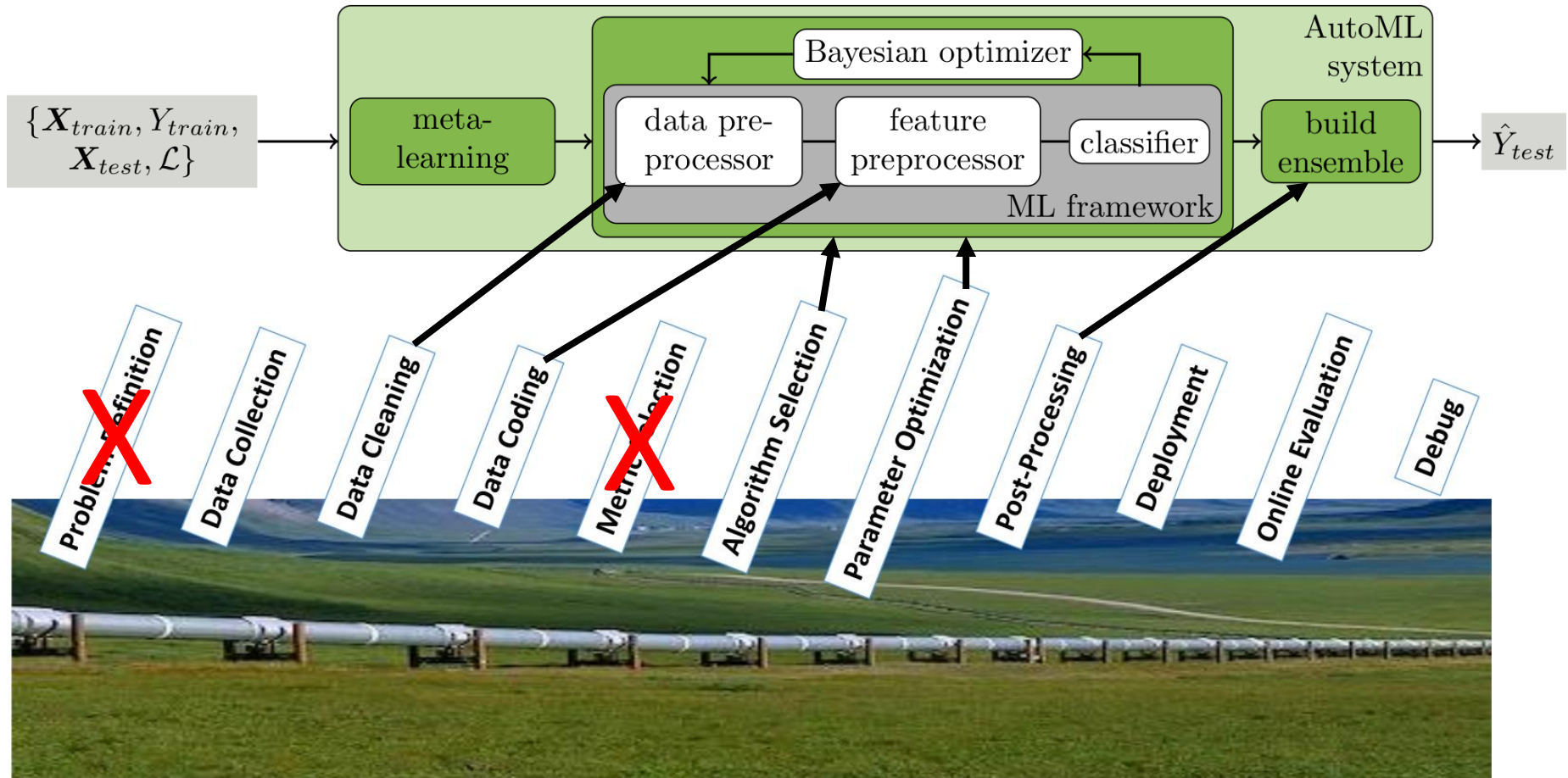


Image credit: Rich Caruana, AutoML 2015

Part 1: General AutoML

1. AutoML by Hyperparameter Optimization
2. Black-box Hyperparameter Optimization
3. Beyond black-box optimization
4. Examples of AutoML
5. **Wrap-up & Conclusion**

HPO for Practitioners: Which Tool to Use?

If you have access to multiple fidelities

- We recommend **BOHB** [Falkner et al, ICML 2018]
- <https://github.com/automl/HpBandSter>
- Combines the advantages of Bayesian optimization and Hyperband

If you do not have access to multiple fidelities

- Low-dim. continuous: GP-based BO
(e.g., BoTorch, MLRMBO, Sigopt, GP version of SMACv3)
- High-dim, categorical, conditional: SMAC or Hyperopt
- Purely continuous, budget >10x dimensionality: CMA-ES

Open-source AutoML Tools based on HPO

- **Auto-WEKA** [\[Thornton et al, KDD 2013\]](#)
 - 768 hyperparameters, 4 levels of conditionality
 - Based on WEKA and SMAC
- **Hyperopt-sklearn** [\[Komer et al, SciPy 2014\]](#)
 - Based on scikit-learn & TPE
- **Auto-sklearn** [\[Feurer et al, NeurIPS 2015\]](#)
 - Based on scikit-learn & SMAC
 - Uses meta-learning and posthoc ensembling
 - Won AutoML competitions 2015-2016 & 2017-2018
- **H2O AutoML** [\[no reference\]](#)
 - Uses implementations from H2O.ai
 - Based on random search and stacking
- **TPOT** [\[Olson et al, EvoApplications 2016\]](#)
 - Based on scikit-learn and evolutionary algorithms
- **ML-PLAN** [\[Mohr et al., Machine Learning 2018\]](#)
 - Based on WEKA and Hierarchical Task Networks

AutoML: Democratization of Machine Learning

Auto-sklearn also won the last two phases of the AutoML challenge [human track \(!\)](#)

- It performed better than up to 130 teams of human experts
- It is open-source (BSD) and trivial to use:


Fork me on GitHub

```
import autosklearn.classification as cls
automl = cls.AutoSklearnClassifier()
automl.fit(X_train, y_train)
y_hat = automl.predict(X_test)
```

automl.github.io/auto-sklearn

 Watch

207

 Star

3,824

 Fork

731

→ More in a hands-on session tomorrow

What have we learned?

1. AutoML by Hyperparameter Optimization

AutoML can be phrased as an HPO problem

2. Black-box Hyperparameter Optimization

We reviewed Bayesian optimization

3. Beyond black-box optimization

Practically applicable by using domain knowledge

4. Meta-learning

Increase practicality by using previous data

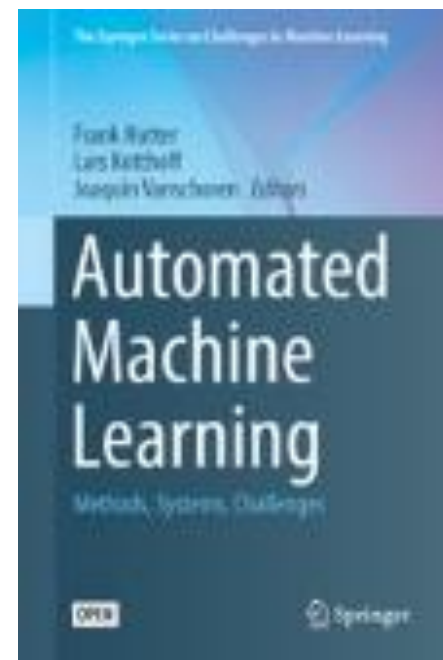
5. Examples

AutoML can be used in almost every step of the ML pipeline

6. Open issues and future work

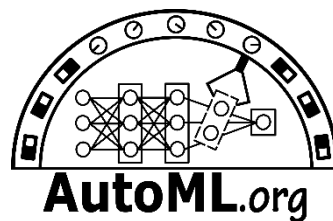
Datasets, search space representation & overfitting

- Automated Machine Learning: Methods, Systems, Challenges
 - Edited by Frank Hutter, Lars Kotthoff and Joaquin Vanschoren
 - Contains introductions to HPO, Meta-Learning and NAS
 - <https://www.springer.com/de/book/9783030053178>
- Various literature reviews on arXiv:
 - [1908.05557](#): Focus on open source software
 - [1810.13306](#): General and comprehensive
 - [1908.00709](#): Focuses mostly on NAS
 - [1905.01392](#): NAS survey
- AutoML workshop video recordings
 - icml2019.automl.org



Thank you for your attention!

Special thanks to Frank Hutter and Joaquin Vanschoren for providing me with the slides this presentation is based on.



Contact:

feurerm@cs.uni-freiburg.de



@__mfeurer__



@automlfreiburg