Oboe: Collaborative Filtering for AutoML Model Selection

Chengrun Yang, Yuji Akimoto, Dae Won Kim, Madeleine Udell

Cornell University

What is AutoML?

an **Automated Machine Learning (AutoML) system**

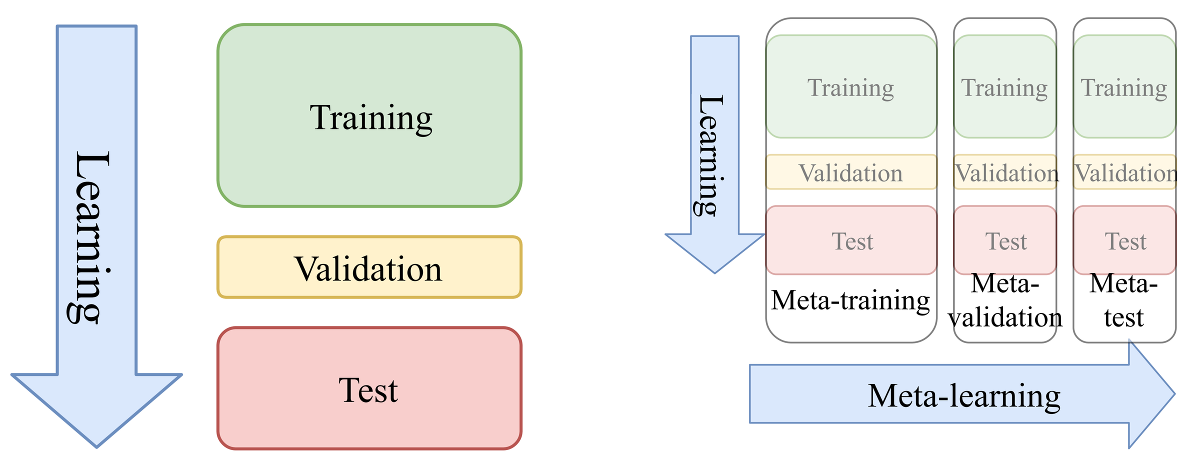
- chooses an algorithm together with hyperparameters
- to achieve the best performance on a (supervised learning) task
- without human intervention.

why AutoML?

- humans are expensive (especially data scientists!)
- computation is cheap
- too many models; can't try them all

to find a reasonable answer, fast, we need:

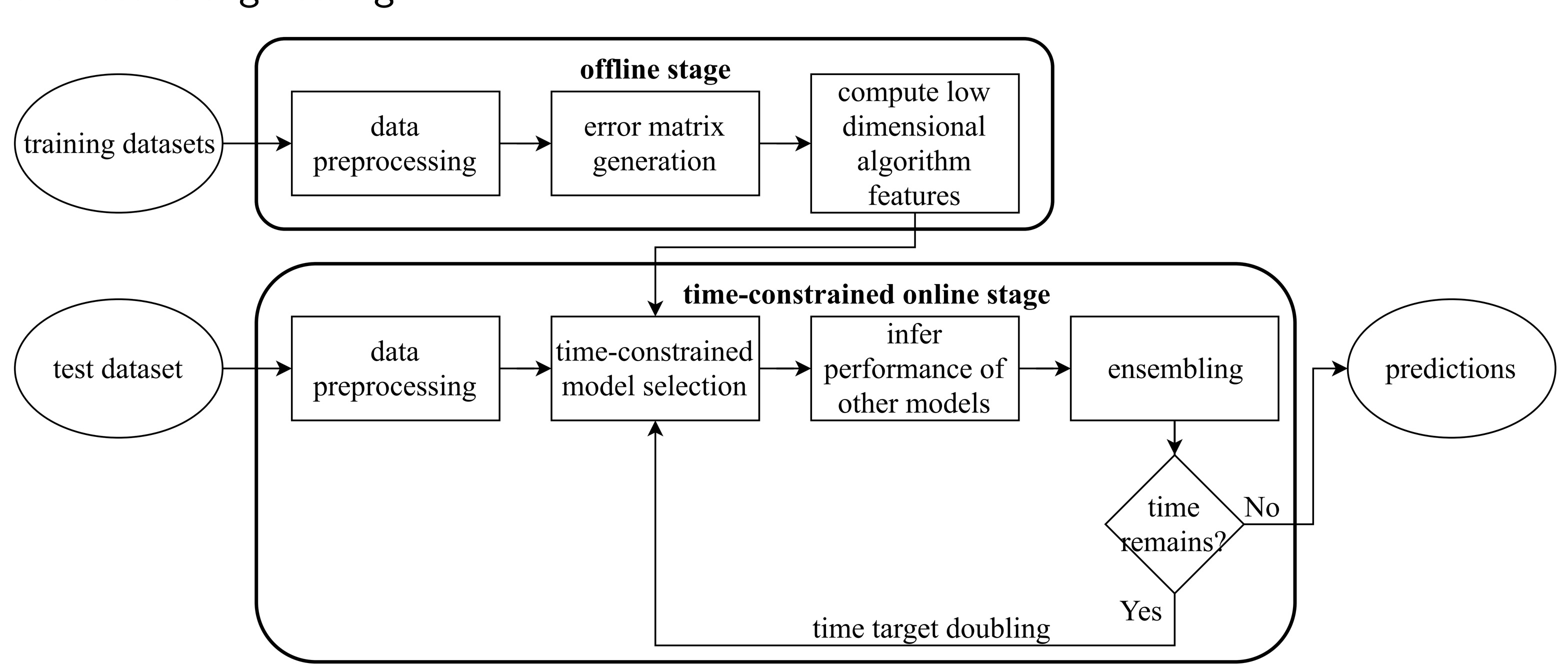
- *Information.* What meta-features predict model performance?
- *Speed.* What meta-features are worth computing?



Our approach

main ideas used by Oboe:

- algorithm performance is **low rank**; rank decomposition gives best meta-features
- use **optimal experiment design** to cold
- the rest is engineering. . .



c.f. SOTA in AutoML: auto-sklearn [2]

at train time (offline stage):

- compute meta-features of training datasets.
- determine best model(s) on training datasets (try them all and pick the best!)

at test time (online stage):

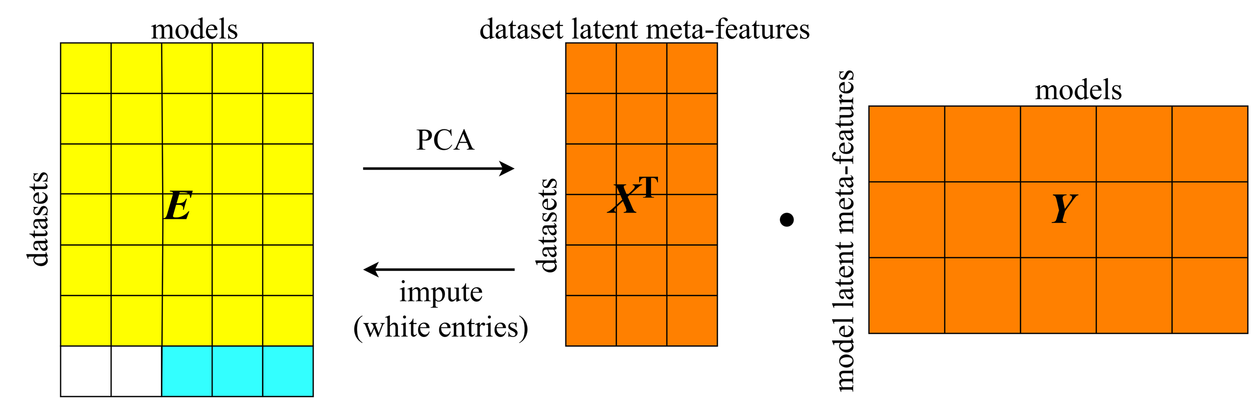
- compute meta-features of test dataset.
- find *similar* datasets (w.r.t. meta-features)
- form ensemble using models that performed best on similar datasets
- tune hyperparameters

e.g., using Gaussian processes [2, 3, 5], bandit-based methods [6], sparse Boolean functions [4] , . . .

AutoML = linear algebra

at train time (offline stage):

- **given:** m training datasets, n machine learning models
- **measure:** error of each model on each dataset
- **form:** $m \times n$ error matrix E (yellow)
- **find:** $X \in \mathbf{R}^{m\times k}$, $Y \in \mathbf{R}^{k\times n}$ (orange) for which



interpretation:

- rows $x_i \in \mathbf{R}^k$ of X are *dataset meta-features*
- columns $y_j \in \mathbf{R}^k$ of Y are *model meta-features*
- $x_i y_j \approx E_{ij}$ are *predicted model performance*

at test time (online stage):

- **given:** new test dataset = new row of E (blue and white)
- **measure:** error of some fast, informative models on new dataset (blue blocks)
- **find:** dataset latent features $\hat{x}$ using least squares
- **compute:** model performance (white blocks) as $\hat{e} = \hat{x}Y$
- **select:** models with best predicted performance to use in ensemble

remaining questions: how to choose rank and find fast, informative models

Experiment design finds fast, informative models

- predict runtime $\hat{t}_j$ of model j on test dataset (predictors = # data points, # features)
- Use (D-optimal) **experiment design** to choose fast, informative models. Solve

$$\begin{aligned} \underset{v_j}{\text{minimize}} \quad & \log\det\Big(\sum_{j=1}^{n} v_j y_j y_j^T\Big)^{-1} \\ \text{subject to} \quad & \sum_{j=1}^{n} v_j \hat{t}_j \le \tau \\ & v_j \in [0,1] \quad \forall j \in [n]. \end{aligned}$$

- Value v_i is large for fast, informative models. Run those! (blue blocks)

Choose a rank you can afford to fit

must run at least k models to fit k-dimensional latent meta-features. . .

given time budget τ for learning on new dataset
initialize rank $k = 1$, time target $t = \tau_0 < \tau/2$
while time remains

- choose k fast, informative models using experiment design
- run those models on the dataset and use to infer performance of all models
- create ensemble using models with predicted best performance
- double time budget t; increase rank k if meta-CV error improves

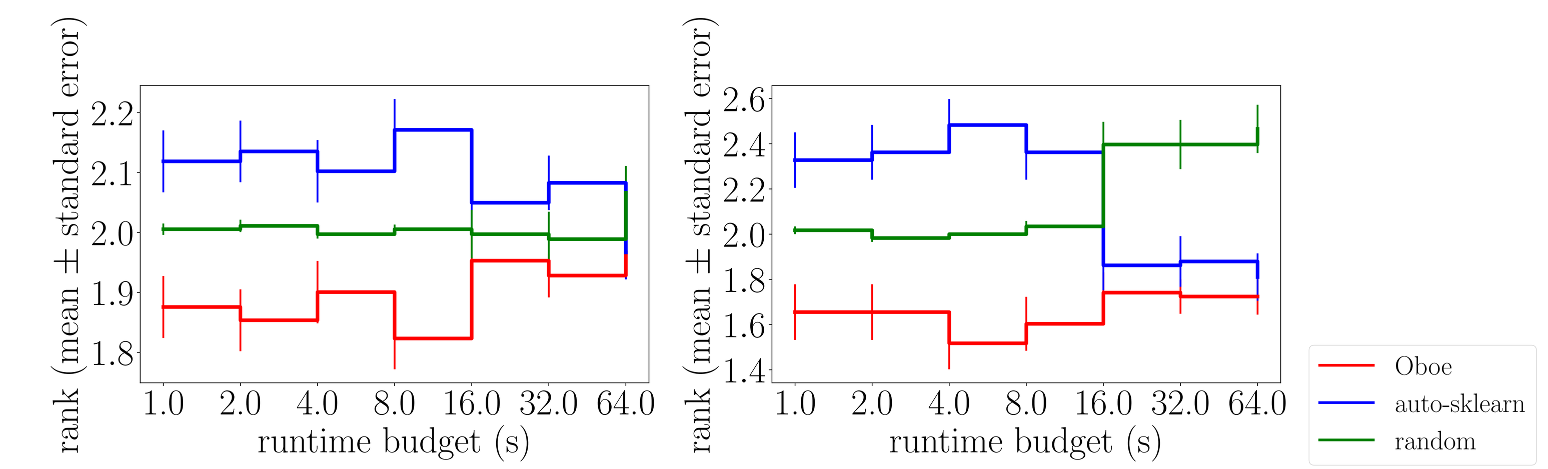
It works!

Experimental setup.

- Datasets: OpenML [8] and UCI [1] datasets with 150–10,000 data points and no missing entries.
- Metric for error matrix: balanced error rate
- Candidate algorithms from python scikit-learn: Adaboost, decision tree, extra trees, random forest, gradient boosting, Gaussian naive Bayes, kNN, logistic regression, multilayer perceptron, perceptron, kernel SVM, linear SVM

Numerical results

- Oboe achieves SOTA performance



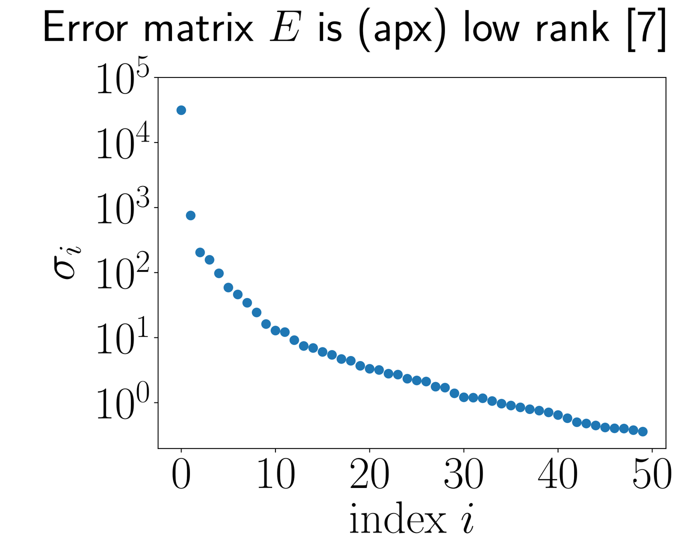
(a) Ranking on OpenML datasets (meta-LOOCV) as a function of time.

(b) Ranking on UCI datasets (meta-test) as a function of time.

- Modeling assumptions are warranted

Algorithm type	Runtime prediction accuracy within factor of 2	Runtime prediction accuracy within factor of 4
Adaboost	83.6%	94.3%
Decision tree	76.7%	88.1%
Extra trees	96.6%	99.5%
Gradient boosting	53.9%	84.3%
Gaussian naive Bayes	89.6%	96.7%
kNN	85.2%	88.2%
Logistic regression	41.1%	76.0%
Multilayer perceptron	78.9%	96.0%
Perceptron	75.4%	94.3%
Random Forest	94.4%	98.2%
Kernel SVM	59.9%	86.7%
Linear SVM	30.1%	73.2%

Error matrix E is (apx) low rank [7]



- Experiment design selects most informative models

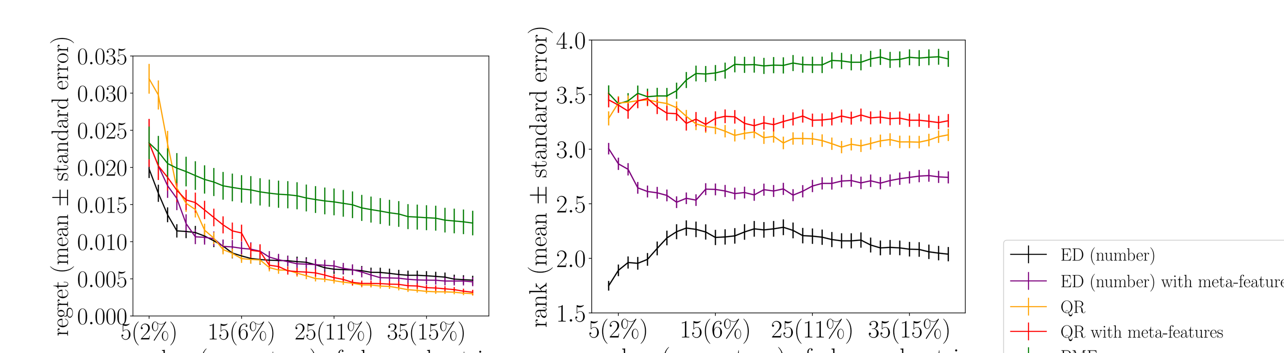


Figure 3: Comparison of sampling schemes (QR or ED) in Oboe and PMF. "QR" denotes QR decomposition with column pivoting; "ED (number)" denotes experiment design with number of observed entries constrained. The left plot shows the regret of each AutoML method as a function of number of entries; the right shows the ranking of each AutoML method in the regret plot (1 is best and 5 is worst).

Thanks!

- Chengrun Yang: cy438@cornell.edu
- Madeleine Udell: udell@cornell.edu

Bibliography

[1] Dua Dheeru and Efi Karra Taniskidou. UCI machine learning repository, 2017.

[2] Matthias Feurer, Aaron Klein, Katharina Eggensperger, Jost Springenberg, Manuel Blum, and Frank Hutter. Efficient and robust automated machine learning. In *Advances in Neural Information Processing Systems*, pages 2962–2970, 2015.

[3] Nicolo Fusi and Huseyn Melih Elibol. Probabilistic matrix factorization for automated machine learning. *Advances in Neural Information Processing Systems*, 2018.

[4] Elad Hazan, Adam Klivans, and Yang Yuan. Hyperparameter Optimization: A Spectral Approach. *arXiv preprint arXiv:1706.00764*, 2017.

[5] Kirthevasan Kandasamy, Willie Neiswanger, Jeff Schneider, Barnabas Poczos, and Eric Xing. Neural Architecture Search with Bayesian Optimisation and Optimal Transport. *Advances in Neural Information Processing Systems*, 2018.

[6] Lisha Li, Kevin Jamieson, Giulia DeSalvo, Afshin Rostamizadeh, and Ameet Talwalkar. Hyperband: A novel bandit-based approach to hyperparameter optimization. *ICLR*, 2017.

[7] Madeleine Udell and Alex Townsend. Why are big data matrices approximately low rank? *SIAM Journal on Mathematics of Data Science*, 1(1):144–160, 2019.