ORIE 4741: Introduction to AutoML

Chengrun Yang

December 3, 2020
About me

- fifth-year PhD student in ECE
- working on AutoML-related problems
- email: cy438@cornell.edu
Outline

Motivation

Some AutoML systems

Demo!

Challenges
Outline

Motivation

Some AutoML systems

Demo!

Challenges
Machine learning is used everywhere ...
But there are real pitfalls ...

1. missing values and outliers are prevalent
2. feature engineering can be misleading
3. model training can be expensive
4. model selection can be more expensive
5. generalization can be tricky
6. ...
A subproblem: estimator selection

hyperparameter: the parameter that governs training process.

Types of hyperparameters: continuous, categorical, ordinal, ...

an estimator: an algorithm with a hyperparameter setting

e.g., ridge regression with $\lambda = 1$, decision tree with depth 3
A subproblem: estimator selection

hyperparameter: the parameter that governs training process. Types of hyperparameters: continuous, categorical, ordinal, ...

an estimator: an algorithm with a hyperparameter setting e.g., ridge regression with $\lambda = 1$, decision tree with depth 3

In supervised learning, given a training set $\{(x_i, y_i)\}_{i=1}^n$, how would we find a mapping $f : \mathcal{X} \rightarrow \mathcal{Y}$?
A subproblem: estimator selection

**hyperparameter**: the parameter that governs training process.
Types of hyperparameters: continuous, categorical, ordinal, ...

**an estimator**: an algorithm with a hyperparameter setting

e.g., ridge regression with \( \lambda = 1 \), decision tree with depth 3

In supervised learning, given a training set \( \{(x_i, y_i)\}_{i=1}^{\ell} \), how would we find a mapping \( f : \mathcal{X} \to \mathcal{Y} \)?

- linear regression?
- random forest?
- gradient boosting?
- ...
- try all the candidates in scikit-learn [PVG+11], or all the available neural network architectures?
The machine learning pipeline space is huge

a **pipeline**: a directed graph of learning components

Data scientists have so many choices to make:

- data imputer: fill in missing values by median? . . .
- encoder: one-hot encode? . . .
- standardizer: rescale each feature? . . .
- dimensionality reducer: PCA, or select by variance? . . .
- estimator: use decision tree or logistic regression? . . .
The machine learning pipeline space is huge

A pipeline: a directed graph of learning components

Data scientists have so many choices to make:

- data imputer: fill in missing values by median? . . .
- encoder: one-hot encode? . . .
- standardizer: rescale each feature? . . .
- dimensionality reducer: PCA, or select by variance? . . .
- estimator: use decision tree or logistic regression? . . .

🤔 In this combinatorially large search space

1. impossible to enumerate all choices on large datasets
2. too expensive on small datasets
3. the best-on-average pipeline does not always perform the best
No Free Lunch

The “no free lunch (NFL)” theorem [Wol96]

There is no one model that works best for every problem.

On 215 midsize OpenML classification datasets:

▶ The best-on-average pipeline (highest average ranking):

- impute missing entries by mode
- encode categorical as integer
- 0 mean and unit variance for each feature
- remove features with 0 variance
- gradient boosting w/ learning rate 0.25 and maximum depth 3

▶ The estimator types of best pipelines on individual datasets:

- gradient boosting - 38.60%
- multilayer perceptron - 20.93%
- kNN - 10.23%
- adaboost - 8.84%
- extra trees - 5.58%
- logistic regression - 5.58%
- decision tree - 3.72%
- random forest - 3.26%
- linear SVM - 1.86%
- Gaussian naive Bayes - 1.40%
Approaches to avoid exhaustive search

1. rule-based search
   - grid search
   - random search
   - genetic programming
   - . . .

2. build *meta-models*!
   - on a single dataset: build *surrogate models* to predict performance of traditional models
     - Gaussian processes
     - reinforcement learning (e.g., multi-armed bandit)
     - experiment design
     - matrix factorization / tensor decomposition
     - . . .
   - learning across datasets, a.k.a. meta-learning
Grid search and random search

On two hyperparameters:

- both are completely uninformed
- random search handles unimportant dimensions better
  - grid search with $M$ explorations on $N$ hyperparameters:
    $\lfloor M^{1/N} \rfloor$ distinct values for each hyperparameter

Image source: Bergstra & Bengio, 2012 [BB12].
Grid search and random search

On two hyperparameters:

- both are completely uninformed
- random search handles unimportant dimensions better
  - grid search with $M$ explorations on $N$ hyperparameters:
    $\lceil M^{1/N} \rceil$ distinct values for each hyperparameter

Poll: the benefit of random search may ___ on a larger number of hyperparameters.

A. increase
B. decrease
C. it depends

Image source: Bergstra & Bengio, 2012 [BB12].
Genetic programming

“Survival of the fittest”: Automatically explore numerous possible pipelines to find the best for the given dataset

Image source: dotnetlovers.com
Bayesian optimization (BO)

BO: a sequential optimization strategy to find the extrema of black-box functions that are expensive to evaluate.

prior + function evaluations = posterior

the most common model: Gaussian processes

Image source: Brochu et al, 2010 [BCDF10].
Multi-armed bandit
Learning vs meta-learning

- learning splits datasets
- meta-learning splits learning instances:
  - same model, different datasets ("sets of datasets")
    e.g., stock market data on different days
  - different models, same dataset
    e.g., performance of ridge regression at different $\lambda$'s
Outline

Motivation

Some AutoML systems

Demo!

Challenges
Some AutoML systems (for reference)

Optimizing over traditional models:

▶ hyperparameter optimization frameworks
▶ **Auto-WEKA** [THHLB13]: Bayesian optimization (BO) on conditional search space
▶ **auto-sklearn** [FKE$^+$15]: meta-learning + BO
▶ **TPOT** [OUA$^+$16]: genetic programming
▶ **Hyperband** [LJD$^+$18]: multi-armed bandit
▶ **PMF** [FSE18]: matrix factorization + BO
▶ **Oboe** [YAKU19]: matrix factorization + experiment design
▶ **AutoGluon** [EMS$^+$20]: ensembling
▶ ...

Neural architecture search (NAS):

▶ **Google NAS** [ZL16]: reinforcement learning
▶ **NASBOT** [KNS$^+$18]: BO + optimal transport
▶ **Auto-Keras** [JSH19]: BO + network morphism
▶ **AutoML-Zero** [RLSL20]: genetic programming
▶ ...

Commercial AutoML tools

- Google AutoML Vision
- Microsoft Azure AutoML
- Amazon AutoGluon on SageMaker
- H2O AutoML
- ...
Outline

Motivation

Some AutoML systems

Demo!

Challenges
Outline

Motivation

Some AutoML systems

Demo!

Challenges
Challenge I: overfitting

Recall overfitting: low training error and high test error
More (layers of) learning, more possible overfitting!

In AutoML,

- **traditional overfitting**: the selected models may overfit on the original dataset
- **meta-overfitting**: the surrogate model may overfit on past learning instances
Challenge II: hyper-hyperparameters

**hyper-hyperparameters:**
hyperparameters of the search rule or meta-model

Example:

- in grid search: selection interval, stopping criteria
- in Gaussian processes: which kernel, kernel parameters, acquisition function, . . .
- in meta-learning: how many datasets to learn from, what models to use for knowledge transfer, . . .

Rationale: make human choices less and easier
Challenge III: robustness

- **traditional robustness**: robustness to noise, outliers, adversarial attacks
- **meta-robustness**: robustness to noisy or adversarial meta-learning instances
A (decisive) impact factor on the advancement of AutoML:
Google RL-based NAS [ZL16]: 1k GPU days (> $70k on AWS)
→ FBNet [WDZ+19]: 10 GPU days ($700 on AWS)
More considerations

- **interpretability**: how to improve it, or do we really need it?
- **baseline**: which one to compare to, human common practice, or human “best” practice?
- **metrics**: how to customize for specific problems?
AutoML has gained popularity in recent years.
People try to automate every phase of machine learning.
Most AutoML frameworks rely on informed search rules or surrogate models.
On top of the challenges in traditional ML, more may arise in AutoML.
James Bergstra and Yoshua Bengio.  
Random search for hyper-parameter optimization.  

Eric Brochu, Vlad M Cora, and Nando De Freitas.  
A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning.  

Nick Erickson, Jonas Mueller, Alexander Shirkov, Hang Zhang, Pedro Larroy, Mu Li, and Alexander Smola.  
Autogluon-tabular: Robust and accurate automl for structured data.  

Matthias Feurer, Aaron Klein, Katharina Eggensperger, Jost Springenberg, Manuel Blum, and Frank Hutter.  
Efficient and robust automated machine learning.  

Nicolo Fusi, Rishit Sheth, and Melih Elibol.  
Probabilistic matrix factorization for automated machine learning.  

Haifeng Jin, Qingquan Song, and Xia Hu.  
Auto-keras: An efficient neural architecture search system.  
Kirthevasan Kandasamy, Willie Neiswanger, Jeff Schneider, Barnabas Poczos, and Eric Xing.
Neural Architecture Search with Bayesian Optimisation and Optimal Transport.

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

Randal S. Olson, Ryan J. Urbanowicz, Peter C. Andrews, Nicole A. Lavender, La Creis Kidd, and
Jason H. Moore.
Applications of Evolutionary Computation: 19th European Conference, EvoApplications 2016, Porto,
Portugal, March 30 – April 1, 2016, Proceedings, Part I, chapter Automating Biomedical Data Science
Through Tree-Based Pipeline Optimization, pages 123–137.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel,
P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher,
M. Perrot, and E. Duchesnay.
Scikit-learn: Machine Learning in Python.

Esteban Real, Chen Liang, David R So, and Quoc V Le.
Automl-zero: Evolving machine learning algorithms from scratch.

Chris Thornton, Frank Hutter, Holger H Hoos, and Kevin Leyton-Brown.
Auto-WEKA: Combined selection and hyperparameter optimization of classification algorithms.
In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data
Fbnet: Hardware-aware efficient convnet design via differentiable neural architecture search.

David H Wolpert.
The lack of a priori distinctions between learning algorithms.

Chengrun Yang, Yuji Akimoto, Dae Won Kim, and Madeleine Udell.
Oboe: Collaborative filtering for automl model selection.

Barret Zoph and Quoc V Le.
Neural architecture search with reinforcement learning.