AutoML: Practical Considerations

Practical and Open Problems

Bernd Bischl Frank Hutter Lars Kotthoff
Marius Lindauer Janek Thomas Joaquin Vanschoren

Choice of Learning Algorithms

- A plethora of learners exists, for different data sets different models are likely needed.
- Studies and experience show:
  One these is often good – on tabular data:
  - Penalized regression, e.g. elastic net
  - Support vector machines
  - Gradient boosting
  - Random forests
  - Neural networks
- Example: Auto-Sklearn 2.0 [Feurer et al. 2020] uses:
  - Extra trees
  - Gradient boosting
  - Passive aggressive
  - Random forest
  - Linear regression with SGD
  - Multi-layer perceptron

Choice of Search Space for a Learning Algorithms

Algorithm	Hyperparameter	Type	Lower	Upper	Trafo
glmnet					
(Elastic net)	alpha	numeric	0	1	-
	lambda	numeric	-10	10	2^x
rpart					
(Decision tree)	cp	numeric	0	1	-
	maxdepth	integer	1	30	-
	minbucket	integer	1	60	-
	minsplit	integer	1	60	-
kknn					
(k-nearest neighbor)	k	integer	1	30	-
svm					
(Support vector machine)	kernel	discrete	-	-	-
	cost	numeric	-10	10	2^x
	gamma	numeric	-10	10	2^x
	degree	integer	2	5	-
ranger					
(Random forest)	num.trees	integer	1	2000	-
	replace	logical	-	-	-
	sample.fraction	numeric	0.1	1	-
	mtry	numeric	0	1	$x \cdot p$
	respect.unordered.factors	logical	-	-	-
	min.node.size	numeric	0	1	n^x
xgboost					
(Gradient boosting)	nrounds	integer	1	5000	-
	eta	numeric	-10	0	2^x
	subsample	numeric	0.1	1	-
	booster	discrete	-	-	-
	max_depth	integer	1	15	-
	min_child_weight	numeric	0	7	2^x
	colsample_bytree	numeric	0	1	-
	colsample_bylevel	numeric	0	1	-
	lambda	numeric	-10	10	2^x
	alpha	numeric	-10	10	2^x

Source: [Probst et al. 2019].

Ranges often selected based on experience

- See other AutoML frameworks: e.g. Auto-Sklearn 2.0 [Feurer et al. 2020]
- Sensitivity analysis often does not exist for ML algorithms
- Check literature on specific ML algorithm

Options for automation:

1. Use huge search space to cover all possibilities (combine with meta-learning for good initial design for Bayesian optimization)
   - Use results of meta-experiments to obtain smaller search space that is estimated to work well.
2. Start with a small space and increase bit by bit

Choice of Resampling Strategy

For computation of generalization error / cost:

$$c(\boldsymbol{\lambda}) = \frac{1}{k}\sum_{i=1}^{k} \widehat{GE}_{\mathcal{D}^i_{\text{val}}}\left(\mathcal{I}(\mathcal{D}^i_{\text{train}}, \boldsymbol{\lambda})\right)$$

Rules of thumb:

- Default: 10-fold CV ($k = 10$)
- Huge datasets: holdout
- Tiny datasets: 10x10 repeated CV
- Stratification for imbalanced classes

Watch out for this:

- Small sample size because of imbalances
- Repeated mesurements (leave-one-object out)
- Time dependencies
- A good AutoML system should let you customize resampling
- Meta-learn good resampling strategy [Feurer et al. 2020]

Choice of Optimization Algorithm

Choose optimization algorithm based on ...

- complexity of search space / budget
- time-costs of evaluations

Complex search space

$\rightarrow$ BO with RF surrogate, EA with exploratory character, TPE

Numerical (lower-dim) search space and tight budget

$\rightarrow$ BO with GP surrogate[1]

Expensive evaluations

$\rightarrow$ Hyperband, Multi-fidelity BO / EAs, ...

Deep learning

$\rightarrow$ common practice: Parameterize architectures, then HPO – better do it jointly!

$\rightarrow$ one-shot models and gradient-based optimization

[1] Still has its own hyperparameters [Lindauer et al. 2019]

HPO Benchmark Suites

HPOBench [Eggensperger et al. 2021].

- Successor of HPOlib
- Collection of 12 benchmark families; in total > 100 HPO problems
- Mix of tabular, surrogate and real benchmark problems
- Also allows for benchmarking multifidelity HPO methods
- Benchmarks are containerized making them easily reproducible

YAHPO Gym [Pfisterer, Schneider et al. 2021].

- Collection of 9 benchmark families constituting over 700 multifidelity multicriteria HPO problems
- Surrogate benchmarks using neural-network based instance surrogates
- fast inference (¡ 50 ms) & low memory footprint ($\sim$ 5 MB)

Others: HPO-B [Arango et al. 2021].

Practical Problems: When to stop?

We need to specify a budget, e.g.

- walltime,
- function evaluations,
- performance threshold, or
- *stagnation* for a certain time.

Problems:

- Overtuning.
- Missed opportunity.
- Wasted computational resources.

Ways out:

- Early stopping for BO [Makarova et al. 2021].
- Rules of thumb, maybe $50 \times d$ to $100 \times d$ (be careful and think for yourself!).
- Expert knowledge.

Practical Problems: Stability

AutoML system should:

- Never fail to return a result.
- Terminate within a given time.
- Save intermediate results and allow to continue.

Failure points:

- Optimizer can crash.
- Pipeline training can crash.
- Training of a pipeline can run "forever".

Ways out:

- Encapsulate train/predict in separate process from HPO.
- Ressource limit time and memory of that process.
- If pipeline crashes, run robust fallback (e.g., constant predictor).
- If optimizer proposal crashes, run random configuration.

Practical Problems: Parallelization

Parallelization should allow:

- Multiple CPUs/GPUs on a single machine.
- Multiple machines / nodes.

Possible parallelization levels:

- Training of pipeline.
- Resampling.
- Evaluation of configurations (batch proposals or asynchronous).

Possible problems:

- Sequential nature of HPO algorithms (e.g. BO).
- Heterogeneous training times of pipelines can cause idling.
- Main memory or CPU-cache becomes bottleneck
- Communication between machine / nodes.

Way out: Use a robust framework for parallelization.

Practical Problems: What to return?

What is the output of a an AutoML system, e.g.

- Pipeline with best validation error.
- Stacking, e.g. averaging, of top-k pipelines.
- Pareto set for multi-objective optimization.
- "One-standard-error rule": Use the simplest model within one standard error of the performance of the best model [Hastie et al. 2009].

Ensure that simple but efficient pipelines have been tried out

- Baseline (Classification: Majority vote; Regression: Mean prediction).
- Linear Model.
- (untuned) Random Forest.
- ...

Interesting Problems

- How to integrate human a-priori knowledge?
- Human-in-the-loop approaches for AutoML.
- How can we best (computationally) transfer “experience” into AutoML?
- Warmstarts, learned search spaces, etc.
- Multi-Objective goals, including model intepretability and fairness.
- AutoML as a process is too much of a black-box, hurts adoption.
- Incorporate Uncertainty quantification into AutoML.
- AutoML beyond supervised learning.
- ...

$\longrightarrow$ Lots of open research questions, feel free to approach us for if you are interested.