AutoML - Benefits, Reality, Future

Koen van der Blom

PGMO Days 2022

2022-11-30
Machine learning

- Widespread
- Automates human work
- Supports human experts
AutoML

Automatically construct (parts of) the machine learning pipeline

Includes, e.g.

- Data processing + cleaning
- Feature construction + selection
- Model selection
- Hyperparameter optimisation
What do we mean when we say AutoML?

Most widespread AutoML tools use meta-algorithms

- Automated algorithm configuration (including HPO + NAS)
- Automated algorithm selection

Many other ideas, but so far quite rare in practice

- Automated data processing, cleaning, etc.
Automated algorithm configuration

Improve performance by finding the best settings/parameters

- Systematic search over the parameter space
- Tries out unconventional parameter combinations
- Usually finds better performing algorithms
  [Fawcett et al. 2011, KhudaBukhsh et al. 2016, Rook et al. 2022]
- Runs while you do other things
Automated algorithm selection

Improve performance by choosing the best algorithm for the job
• Predict which algorithm to use for each problem instance
• Algorithm selection usually outperforms the single best solver
Benefits of AutoML

Performance improvement over hand designed systems

Democratise ML

• Reduce workload for ML experts
• Reduce required ML expertise

Dream: Create an ML system with one press on the button
Great benefits, great adoption?

Study AutoML adoption in software engineering for ML
  • Systems where ML is used in real applications

Questionnaire [Serban et al. 2020 + 2021]
  • Measure adoption levels of various AutoML techniques

Interviews [KvdB et al. 2021]
  • Insights into reasons for adoption and benefits in practice

Joint work with Alex Serban, Joost Visser & Holger Hoos
AutoML adoption is not as high as expected!

- 20-30% do not adopt AutoML at all
- Another 50-60% do not completely adopt AutoML
Why is AutoML adoption lower than expected?

Points mentioned by two interviewees

- High initial cost to adoption (missing expertise)
- Difficult to predict good run length for AutoML
- Unclear what is wrong when AutoML systems fail
- Limited availability of computational resources

Literature also suggests usability, interpretability, interactivity
What to do?

Improved AutoML systems are needed

Start with the basics

- AutoML tools for computer scientists without AutoML expertise
- Meta algorithms
  - Automated algorithm configuration
  - Automated algorithm selection
Sparkle: Accessible meta-algorithms

Lower the bar to use algorithm selection and configuration

• Ease of use: Automate where possible
• Correctness: Implement best practices, avoid pitfalls
• Explanation: Clear commands, detailed (but concise) reporting

Sparkle platform: https://bitbucket.org/sparkle-ai/sparkle/  [KvdB et al. 2022]

Joint work with Holger Hoos, Chuan Luo, Jeroen Rook
Algorithm selector construction

Prepare algorithm, feature extractor, wrappers, instances, selector constructor, . . .

1. Prepare performance data
   a. Provide target algorithm executables
   b. Manage target algorithm cutoff time
      Wrapper, Sparkle code
   c. Output performance data in selector constructor format
      Sparkle code

2. Problem instances
   a. Assemble set of instances
   b. Split into train and test

3. Prepare feature data
   a. Provide feature extractor executable(s)
   b. Manage feature computation cutoff time
      Wrapper, Sparkle code
   c. Output feature data in selector constructor format
      Sparkle code

4. Selector construction procedure
   a. Choose which selector construction procedure to use
   b. Install selector construction procedure
      Sparkle installation already includes installation of the selector construction procedure

5. Scenario
   a. Set performance measure
   b. Set target algorithm cutoff time

6. Selector construction protocol
   a. Set selector construction budget
   b. Validate on training set
      Sparkle ensures the produced selector is compared against the single best solver and the perfect selector

7. Interpret and write up results
   a. Interpret raw output
      Sparkle extracts the most important information from the output files and presents it in a LaTeX/PDF report
   b. Calculate marginal contribution of each solver
      Sparkle computes the relative marginal contribution of each solver with regard to the final constructed selector
   c. Write up results
      Sparkle describes the experimental procedure and provides references for used tools in its report
Algorithm configuration

Prepare algorithm, wrappers, parameter space, instances, configurator, ...

1. Target algorithm
   a. Provide executable

2. Solver wrapper
   a. Handle configurator call format for: Instance, parameter settings, random seed
   b. Output (compute) cost metric in configurator format

3. Configuration space
   a. Set parameter values/ranges
   b. Set parameter default values
   c. Indicate conditional and invalid parameter combinations

4. Problem instances
   a. Assemble homogeneous set of instances
   b. Split into train and test
   c. Create file with instance paths
      Sparkle generates these for provided directories containing instance sets

5. Configurator
   a. Choose which configurator to use
   b. Install configurator
      Sparkle installation already includes installation of the configurator

6. Scenario
   a. Set performance measure
   b. Set target algorithm cutoff time

Algorithm configuration procedure & execution

7. Configuration protocol
   a. Set configuration budget (per run)
   b. Set number of configuration runs
      Sparkle defaults to multiple runs
   c. Create configuration scenario file
      Sparkle generates this based on input
   d. Validate on training set for each run
      Sparkle ensures validation is performed
   e. Choose best over all runs
      Sparkle ensures the best configuration (after validation on train) is selected
   f. Validate on the testing set
      Sparkle ensures only the final configuration is compared on the testing set

Result interpretation & analysis

8. Interpret and write up results
   a. Interpret raw output
      Sparkle extracts the most important information from the output files and presents it in a \LaTeX/PDF report
   b. Compare optimised configuration with default
      Sparkle includes a visual comparison in its report, in addition to the performance values
   c. Write up results
      Sparkle describes the experimental procedure and provides references for used tools in its report
Command example

1: Commands initialise.py
2: Commands add_instances.py Resources/PTN/
3: Commands add_instances.py Resources/PTN2/
4: Commands add Solver.py --deterministic 0 Resources/PbO-CSCCSAT/
5: Commands configure Solver.py --solver PbO-CSCCSAT --instance-set-train PTN
6: Commands sparkle wait.py
7: Commands validate configured vs default.py --solver Solvers/PbO-CCSAT-Generic/ --instance-set-train Instances/PTN/ --instance-set-test Instances/PTN2/
8: Commands generate_report.py
Reporting

Often very basic in meta-algorithm tools (even just performance)

Detailed but concise report in Sparkle

- Used instance sets, target algorithm, configurator/selector
- Experiment description (protocol, budgets, . . . )
- Performance values + plots
Future of AutoML

No ‘one button’ magic system

Instead AutoML tools that support ML experts (eventually laymen)
- Correctness
- Understandability
- Interpretability
- Interaction

A lot of work needed beyond Sparkle!
Take away

- AutoML has the potential to democratise ML
- Only 20-30% adopt AutoML, another 50-60% only partially
- Adoption held back by missing expertise, usability, etc.
- Sparkle: Make meta-algorithms (core AutoML tools) accessible
- Future: Need tools for other audiences + AutoML components

Sparkle platform: https://bitbucket.org/sparkle-ai/sparkle/

koen.vdblom@lip6.fr
References


