AI-Augmented Algorithms – How I Learned to Stop Worrying and Love Choice

Lars Kotthoff

University of Wyoming
larsko@uwyo.edu

Glasgow, 23 July 2018
Outline

▷ Big Picture
▷ Motivation
▷ Algorithm Selection and Portfolios
▷ Algorithm Configuration
▷ Outlook
Big Picture

▷ advance the state of the art through meta-algorithmic techniques
▷ rather than inventing new things, use existing things more intelligently – automatically
▷ invent new things through combinations of existing things
Big Picture

▷ advance the state of the art through meta-algorithmic techniques
▷ rather than inventing new things, use existing things more intelligently – automatically
▷ invent new things through combinations of existing things
Motivation – What Difference Does It Make?
Performance Improvements

Performance models of black-box processes

▷ also called surrogate models
▷ replace expensive underlying process with cheap approximate model
▷ build approximate model based on real evaluations using machine learning techniques
▷ no knowledge of what the underlying process does required (but can be helpful)
▷ allow better understanding of the underlying process through interrogation of the model
Algorithm Selection
Algorithm Selection

Given a problem, choose the best algorithm to solve it.

Algorithm Selection

Portfolio
- Algorithm 1
- Algorithm 2
- Algorithm 3

Training Instances
- Instance 1
- Instance 2
- Instance 3

Feature Extraction

Algorithm Selection

Performance Model

Instance 4: Algorithm 2
Instance 5: Algorithm 3
Instance 6: Algorithm 3

Feature Extraction

Instance 4
Instance 5
Instance 6
...
Algorithm Portfolios

▷ instead of a single algorithm, use several complementary algorithms
▷ idea from Economics – minimise risk by spreading it out across several securities
▷ same for computational problems – minimise risk of algorithm performing poorly
▷ in practice often constructed from competition winners

“algorithm” used in a very loose sense
▷ algorithms
▷ heuristics
▷ machine learning models
▷ consistency levels
▷ ...

Algorithms
Parallel Portfolios

Why not simply run all algorithms in parallel?

▷ not enough resources may be available/waste of resources
▷ algorithms may be parallelized themselves
▷ memory contention
Building an Algorithm Selection System

- most approaches rely on machine learning
- train with representative data, i.e. performance of all algorithms in portfolio on a number of instances
- evaluate performance on separate set of instances
- potentially large amount of prep work
Key Components of an Algorithm Selection System

▷ feature extraction
▷ performance model
▷ prediction-based selector/scheduler

optional:
▷ presolver
▷ secondary/hierarchical models and predictors (e.g. for feature extraction time)
Types of Performance Models

- **Classification Model**
  - A1 vs. A2
  - A1 vs. A3
  - A1: 1 vote
  - A2: 0 votes
  - A3: 2 votes

- **Regression Models**
  - A1: 1.2
  - A2: 4.5
  - A3: 3.9

- **Pairwise Classification Models**
  - A1 - A2
  - A1 - A3
  - A1: 1 vote
  - A2: 0 votes
  - A3: 2 votes

- **Pairwise Regression Models**
  - A1 - A2
  - A1 - A3
  - A1: -1.3
  - A2: 0.4
  - A3: 1.7

Instance 1: Algorithm 2
Instance 2: Algorithm 1
Instance 3: Algorithm 3...
Benchmark Library – ASlib

▷ currently 29 data sets/scenarios with more in preparation
▷ SAT, CSP, QBF, ASP, MAXSAT, OR, machine learning...
▷ includes data used frequently in the literature that you may want to evaluate your approach on
▷ performance of common approaches that you can compare to
▷ http://aslib.net

_____________________________________________________


http://larskotthoff.github.io/assurvey/
Algorithm Configuration
Algorithm Configuration

Given a (set of) problem(s), find the best parameter configuration.
Parameters?

▷ anything you can change that makes sense to change
▷ e.g. search heuristic, variable ordering, type of global constraint decomposition
▷ not random seed, whether to enable debugging, etc.
▷ some will affect performance, others will have no effect at all
Automated Algorithm Configuration

▷ no background knowledge on parameters or algorithm
▷ as little manual intervention as possible
  ▷ failures are handled appropriately
  ▷ resources are not wasted
  ▷ can run unattended on large-scale compute infrastructure
Algorithm Configuration

Parameter domains & starting values

Configurator

Calls with different parameter settings

Configuration scenario

Target algorithm

Solves

Problem instances

Returns solution cost

General Approach

▷ evaluate algorithm as black box function
▷ observe effect of parameters without knowing the inner workings
▷ decide where to evaluate next
▷ balance diversification/exploration and intensification/exploitation
When are we done?

- most approaches incomplete
- cannot prove optimality, not guaranteed to find optimal solution (with finite time)
- performance highly dependent on configuration space

→ How do we know when to stop?
Time Budget

How much time/how many function evaluations?
▷ too much → wasted resources
▷ too little → suboptimal result
▷ use statistical tests
▷ evaluate on parts of the instance set
▷ for runtime: adaptive capping
Grid and Random Search

▷ evaluate certain points in parameter space

---

Model-Based Search

▷ evaluate small number of configurations
▷ build model of parameter-performance surface based on the results
▷ use model to predict where to evaluate next
▷ repeat
▷ allows targeted exploration of new configurations
▷ can take instance features into account like algorithm selection

Model-Based Search Example

Iter = 1, Gap = 1.9909e−01
Model-Based Search Example
Model-Based Search Example

Iter = 3, Gap = 1.9909e-01
Model-Based Search Example
Model-Based Search Example

 Iter = 6, Gap = 1.9996e-01
Model-Based Search Example

Iter = 7, Gap = 2.0000e−01

- $y$ and $y_{hat}$ plotted as solid and dashed lines, respectively.
- $e_i$ plotted on the lower graph.
- Points labeled as 'init', 'prop', and 'seq' indicated.
- Axes labeled $x$ and $y$.
- Scale for $y$ and $e_i$.

Parameters:
- $x$ range: $-1.0$ to $1.0$
- $y$ range: $0.0$ to $0.8$
- $e_i$ range: $0.0e+00$ to $5e−05$
Model-Based Search Example

Iter = 8, Gap = 2.0000e−01

- Iteration 8
- Gap = 0.2000

- Type: init, prop, seq
- Y values: 0, 0.4, 0.8
- X values: -1.0, -0.5, 0.0, 0.5, 1.0
- Error values: 0.0e+00, 5.0e−06, 1.0e−05, 1.5e−05, 2.0e−05

Graph showing the model-based search example with iterations and gap values.
Model-Based Search Example
Model-Based Search Example

Iter = 10, Gap = 2.0000e−01
Benchmark Library – AClib

▷ ASP, MIP, planning, machine learning, ...
▷ 4 algorithm configuration tools from the literature already integrated
▷ https://bitbucket.org/mlindauer/aclib2

---

Outlook
Quo Vadis, Software Engineering?
Quo Vadis, Software Engineering?

Meta-Algorithmics in the Physical Realm – AI and Lasers
Tools and Resources

**LLAMA**  [https://bitbucket.org/lkotthoff/llama](https://bitbucket.org/lkotthoff/llama)
**SATzilla**  [http://www.cs.ubc.ca/labs/beta/Projects/SATzilla/](http://www.cs.ubc.ca/labs/beta/Projects/SATzilla/)
  **iRace**  [http://iridia.ulb.ac.be/irace/](http://iridia.ulb.ac.be/irace/)
**mlrMBO**  [https://github.com/mlr-org/mlrMBO](https://github.com/mlr-org/mlrMBO)
**Spearmint**  [https://github.com/HIPS/Spearmint](https://github.com/HIPS/Spearmint)
**TPE**  [https://jaberg.github.io/hyperopt/](https://jaberg.github.io/hyperopt/)

**autofolio**  [https://bitbucket.org/mlindauer/autofolio/](https://bitbucket.org/mlindauer/autofolio/)
**Auto-sklearn**  [https://github.com/automl/auto-sklearn](https://github.com/automl/auto-sklearn)
Summary

Algorithm Selection choose the best *algorithm* for solving a problem

Algorithm Configuration choose the best *parameter configuration* for solving a problem with an algorithm

▷ mature research areas
▷ can combine configuration and selection
▷ effective tools are available
▷ COnfiguration and SElection of ALgorithms group COSEAL
  http://www.coseal.net

Don’t set parameters prematurely, embrace choice!
I’m hiring!

Several funded graduate positions available.