Algorithm Selection
Predict which algorithm to use!
The Problem
Availability of Algorithms

Machine Learning
- K-nearest Neighbor
- SVM
- Random Forest
- Gradient Boosting
- Deep Neural Network

Satisfiability Solving
- lingeling
- cryptominisat
- glucose
- probSAT
- CaDiCaL
- syrup

Sorting
- Merge Insertion
- Quick sort
- Merge sort
- Binary tree sort
Manual Algorithm Selection

**Goal:** Select the algorithm with the best performance for a given instance.
Algorithm Selection Matters!

Running always the same algorithm
Selecting the best algorithm for each instance!

Average Running Time

10  50  100  500  1000  5000

Single Algorithm
Optimal Algorithm
## Open Algorithm Selection Challenge 2017

[Lindauer et al. AIJ 2019]

<table>
<thead>
<tr>
<th>Domain</th>
<th>Average opt. Speedup</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mixed Integer Programming (MIP)</td>
<td>10</td>
</tr>
<tr>
<td>Maximum Satisfiability Problem (MAXSAT)</td>
<td>15</td>
</tr>
<tr>
<td>Boolean Satisfiability Problem (SAT)</td>
<td>30</td>
</tr>
<tr>
<td>Structure learning in Bayesian networks</td>
<td>41</td>
</tr>
<tr>
<td>Constraint Satisfaction Problem (CSP)</td>
<td>61</td>
</tr>
<tr>
<td>Quantified Boolean Formula (QBF)</td>
<td><strong>264</strong></td>
</tr>
<tr>
<td>Machine Learning (OPENML-Weka; absolute impr.)</td>
<td>2%</td>
</tr>
</tbody>
</table>

Data available in **ASlib** [Bischl, Lindauer et al. AIJ 2016]
Manual Algorithm Selection

Portfolio of Algorithms

Instance → Solve instance

Goal: Select the algorithm with the best performance for a given instance
Can we automate Algorithm Selection? [Rice’76]

**Goal:** Predict the algorithm with the best performance for a given instance

Instance → Numerical Representation → Predictions via Machine Learning → Portfolio of Algorithms
Instance Features = Numerical Representations

● Counting Features
  ○ How large/hard is the instance?
    ■ Examples: #variables, #constraints, #data points, #list entries, …

● Probing Features
  ○ Run a simple algorithm to check behavior
    ■ Examples: accuracy of decision tree, performance of local search SAT solver, …

● Important properties of instance features
  ○ Informative about performance of algorithms
  ○ cheap-to-compute
Algorithm Selection [Rice’76]

Goal: Predict the algorithm with the best performance for a given instance
## Collecting Data

### Training performance data

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>5</td>
<td>88</td>
<td>10</td>
<td>9</td>
<td></td>
</tr>
<tr>
<td>42</td>
<td>24</td>
<td>30</td>
<td>32</td>
<td><strong>10</strong></td>
<td></td>
</tr>
<tr>
<td>488</td>
<td>888</td>
<td><strong>55</strong></td>
<td>487</td>
<td>87</td>
<td></td>
</tr>
<tr>
<td>452</td>
<td>123</td>
<td>188</td>
<td><strong>78</strong></td>
<td>480</td>
<td></td>
</tr>
<tr>
<td>102</td>
<td>488</td>
<td>300</td>
<td>330</td>
<td><strong>51</strong></td>
<td></td>
</tr>
<tr>
<td><strong>1</strong></td>
<td><strong>18</strong></td>
<td><strong>6</strong></td>
<td><strong>8</strong></td>
<td><strong>88</strong></td>
<td></td>
</tr>
</tbody>
</table>

### Unknown test data

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>??</td>
<td>??</td>
<td>??</td>
<td>??</td>
<td>??</td>
<td>??</td>
</tr>
<tr>
<td>??</td>
<td>??</td>
<td>??</td>
<td>??</td>
<td>??</td>
<td>??</td>
</tr>
<tr>
<td>??</td>
<td>??</td>
<td>??</td>
<td>??</td>
<td>??</td>
<td>??</td>
</tr>
<tr>
<td>??</td>
<td>??</td>
<td>??</td>
<td>??</td>
<td>??</td>
<td>??</td>
</tr>
<tr>
<td>??</td>
<td>??</td>
<td>??</td>
<td>??</td>
<td>??</td>
<td>??</td>
</tr>
<tr>
<td>??</td>
<td>??</td>
<td>??</td>
<td>??</td>
<td>??</td>
<td>??</td>
</tr>
</tbody>
</table>

→ Check out aslib.net
Algorithm Selection [Rice’76]

Task

Numerical Representation

Portfolio of Algorithms

Predictions via Machine Learning

Goal: Predict the algorithm with the best performance for a given instance
Approaches
Algorithm Selection: Idea #1 [Kadiolgu et al. 2010]

**Idea:** Similar instances should be assigned to the same algorithm

- Human-inspired strategy
- 1. cluster instances
- 2. Assign best algorithm in each cluster
Algorithm Selection: Idea #1 [Kadiolgu et al. 2010]

- Very easy to implement
- Only a single model
- Very fast to train model

- Unsupervised learning
  → clusters could be wrong
- Typically worse performance than other approaches
Algorithm Selection \cite{Rice’76}

Goal: *Predict* the algorithm with the best **performance** for a given instance
**Algorithm Selection: Idea #2** [Xu et al. 2010]

**Idea:** Predict the performance of each algorithm and select the best performing one.
Algorithm Selection: Idea #2 [Xu et al. 2010]

- Easy to implement
- Supervised learning
- Can be used for more than algorithm selection

- Training of $n$ models for $n$ algorithms
- Learns a harder task than necessary
Algorithm Selection

[Reference: Rice’76]

**Goal:** Predict the algorithm with the best performance for a given instance
**Idea:** Learn a classification model for each pair of algorithms

<table>
<thead>
<tr>
<th>Label</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>Very important instances</td>
<td>400</td>
</tr>
<tr>
<td>Less important instances</td>
<td>5</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Label</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>42</td>
<td>24</td>
<td>18</td>
</tr>
<tr>
<td>488</td>
<td>888</td>
<td>400</td>
</tr>
<tr>
<td>452</td>
<td>123</td>
<td>329</td>
</tr>
<tr>
<td>102</td>
<td>488</td>
<td>386</td>
</tr>
<tr>
<td>1</td>
<td>18</td>
<td>17</td>
</tr>
</tbody>
</table>
**Algorithm Selection: Idea #3** [Xu et al. 2011]

**Idea:** For a new instance, use a voting scheme on pairwise predictions

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>0</th>
<th>1</th>
<th>0</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td></td>
<td>1</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

#Votes

- 1
- 4
- 2
- 0
- 3

Selected algorithm: 4 votes
Algorithm Selection: Idea #3 [Xu et al. 2010]

- Supervised learning
- Weighting of instances
- State-of-the-art approach

- Training of $n^2/2$ models for $n$ algorithms
- Predictions from $n^2/2$ models
Algorithm Selection [Rice’76]

Goal: Predict the algorithm with the best performance for a given instance
**Idea:** Learn pairwise regressions model for difference in performance

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th>Performance difference</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>5</td>
<td></td>
<td>5</td>
</tr>
<tr>
<td>42</td>
<td>24</td>
<td></td>
<td>18</td>
</tr>
<tr>
<td>488</td>
<td>888</td>
<td></td>
<td>-400</td>
</tr>
<tr>
<td>452</td>
<td>123</td>
<td></td>
<td>329</td>
</tr>
<tr>
<td>102</td>
<td>488</td>
<td></td>
<td>-386</td>
</tr>
<tr>
<td>1</td>
<td>18</td>
<td></td>
<td>-17</td>
</tr>
</tbody>
</table>
**Idea:** For a new instance, sum up differences in performance predictions

<table>
<thead>
<tr>
<th></th>
<th>-2</th>
<th>-5</th>
<th>-100</th>
<th>50</th>
<th>Advantage</th>
</tr>
</thead>
<tbody>
<tr>
<td>-</td>
<td></td>
<td></td>
<td>-100</td>
<td>50</td>
<td>-57</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>1</td>
<td>11</td>
<td>5</td>
<td>19</td>
</tr>
<tr>
<td>5</td>
<td>-1</td>
<td></td>
<td>100</td>
<td>-5</td>
<td>99</td>
</tr>
<tr>
<td>100</td>
<td>-11</td>
<td>-100</td>
<td></td>
<td>-12</td>
<td>-23</td>
</tr>
<tr>
<td>-50</td>
<td>-5</td>
<td>5</td>
<td>12</td>
<td></td>
<td>-38</td>
</tr>
</tbody>
</table>

Most Votes: 19

Selected algorithm: -100

[Keythoff 2015]
Algorithm Selection: Idea #4 [Kotthoff 2015]

- Supervised learning
- Takes performance difference in labels into account

- Training of \( n^2/2 \) models for \( n \) algorithms
- Predictions from \( n^2/2 \) models
Overview of Algorithm Selection Approaches

Clustering

Regression

Pairwise Classification

Pairwise Regression

Predictions

Predictions
Automate
Automated Algorithm Selection
## Comparison of Algorithm Selection Approaches

### Insight:
Different applications require different selection approaches!

<table>
<thead>
<tr>
<th>Applications</th>
<th>Selection Tools</th>
<th>3S-like</th>
<th>aspeed</th>
<th>claspholio-1.0-like</th>
<th>ISAC-like</th>
<th>ME-ASP-like</th>
<th>SATzilla'09-like</th>
<th>SATzilla'11-like</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASP-POTASSCO</td>
<td>4.1</td>
<td>1.4</td>
<td>2.8</td>
<td>3.8</td>
<td>1.9</td>
<td>2.9</td>
<td>4.2</td>
<td></td>
</tr>
<tr>
<td>CSP-2010</td>
<td>1.5</td>
<td>1.0</td>
<td>2.1</td>
<td>2.1</td>
<td>2.6</td>
<td>2.5</td>
<td>3.1</td>
<td></td>
</tr>
<tr>
<td>MAXSAT12-PMS</td>
<td>6.5</td>
<td>2.7</td>
<td>1.6</td>
<td>4.9</td>
<td>2.1</td>
<td>3.4</td>
<td>8.6</td>
<td></td>
</tr>
<tr>
<td>PREMARSHALLING</td>
<td>2.9</td>
<td>3.6</td>
<td>1.2</td>
<td>1.3</td>
<td>1.1</td>
<td>1.5</td>
<td>2.3</td>
<td></td>
</tr>
<tr>
<td>PROTEUS-2014</td>
<td>10.9</td>
<td>6.3</td>
<td>3.5</td>
<td>4.3</td>
<td>3.1</td>
<td>4.9</td>
<td>6.5</td>
<td></td>
</tr>
<tr>
<td>QBF-2011</td>
<td>7.7</td>
<td>4.9</td>
<td>2.3</td>
<td>2.8</td>
<td>2.8</td>
<td>3.7</td>
<td>9.8</td>
<td></td>
</tr>
<tr>
<td>SAT11-HAND</td>
<td>2.6</td>
<td>3.6</td>
<td>1.1</td>
<td>1.2</td>
<td>1.0</td>
<td>1.9</td>
<td>2.3</td>
<td></td>
</tr>
<tr>
<td>SAT11-INDU</td>
<td>1.2</td>
<td>1.1</td>
<td>1.2</td>
<td>1.3</td>
<td>1.2</td>
<td>1.1</td>
<td>1.2</td>
<td></td>
</tr>
<tr>
<td>SAT11-RAND</td>
<td>3.9</td>
<td>4.7</td>
<td>1.2</td>
<td>2.5</td>
<td>1.8</td>
<td>2.6</td>
<td>3.8</td>
<td></td>
</tr>
<tr>
<td>SAT12-ALL</td>
<td>1.5</td>
<td>1.1</td>
<td>1.2</td>
<td>1.1</td>
<td>1.1</td>
<td>1.4</td>
<td>1.8</td>
<td></td>
</tr>
<tr>
<td>SAT12-HAND</td>
<td>1.7</td>
<td>1.8</td>
<td>1.1</td>
<td>1.1</td>
<td>1.0</td>
<td>1.5</td>
<td>1.9</td>
<td></td>
</tr>
<tr>
<td>SAT12-INDU</td>
<td>1.2</td>
<td>0.8</td>
<td>1.2</td>
<td>1.2</td>
<td>1.1</td>
<td>1.3</td>
<td>1.3</td>
<td></td>
</tr>
<tr>
<td>SAT12-RAND</td>
<td>0.8</td>
<td>0.8</td>
<td>0.6</td>
<td>0.9</td>
<td>0.9</td>
<td>0.9</td>
<td>0.9</td>
<td></td>
</tr>
<tr>
<td>geo. mean</td>
<td>2.6</td>
<td>2.0</td>
<td>1.5</td>
<td>1.9</td>
<td>1.5</td>
<td>2.0</td>
<td>2.8</td>
<td></td>
</tr>
</tbody>
</table>
Challenges in Applying Algorithm Selection

- For each application, we potentially need a different approach
  - clustering vs. regression vs. pairwise classification vs. pairwise regression
- Each approach can be implemented with different machine learning algorithms
  - Random forest, SVM, deep neural network, gradient boosting
- Each machine learning algorithm requires optimal hyperparameter settings
  - Kernel width of SVM?
  - Pruning strength of trees?
  - ...

→ Effective application of algorithm selection in practice can be hard!
Algorithm Selection Design Choices
AutoML saves the day!

- Insight: Algorithm selection is yet another machine learning problem (with a special design spaces)

- Automated machine learning:
  - Automated search for best machine learning algorithm and its hyperparameter settings
  - Allows for automated deployment of algorithm selection in practice
Crash course: AutoML

Machine Learning Algorithm

Design Options

select

Performance (e.g., RMSE)

π

\( \Delta \)

42

X

Black-Box

f(x)
Crash course: AutoML

Machine Learning Algorithm

\[ f(x) \]

Design Options

X

Performance (e.g., RMSE)

- Not very efficient
- Error-prone
- Requires expert knowledge

select

...
Crash course: AutoML

- Trade-off
- Exploration-Exploitation
- Data efficient
- State-of-the-art

Not easy to parallelize
AutoFolio: Algorithm Selection + AutoML

[Lindauer et al. 2015]
AutoFolio on SAT [Lindauer et al. 2015]

Legend:
- Dot: one instance
- Metric: runtime
- Portfolio: set of SAT solvers
- x-axis: default selection approach
- y-axis: optimized selection approach
AutoFolio [Lindauer et al. 2015]

5 - 10 fold speedup!
(on these examples)
Insights from AutoFolio [Lindauer et al. 2015]

- Most important design decision: 
  *How much time do I invest in instance feature computation?*
- 2nd most important design decision: 
  *Algorithm selection approach*
Extensions of Algorithm Selection
Schedules of Algorithms

**Idea:** Instead of a single algorithm, we want to run several algorithms in a sequence.
**Idea #1: Pre-solving schedules** [Xu et al. 2010]

**Idea:** First runs a static schedule of algorithms (independent of given instance). If it fails, use algorithm selection (based on instance features).

**Challenge:** Find a well-performing schedule → hard optimization-problem

**Insight** [Gonard et al. 2016]: Pre-solving schedule has to take into account prediction model and vice versa.
Idea #2: Predict schedule of algorithms

Idea: Predict a schedule of algorithms once in the beginning. The schedule is instance-dependent.

Idea [Amadini et al. 2014]: Sort algorithms by probability to solve given instance.
Idea #3: Sequential Predictions

**Idea:** Predict an algorithm and update your believe based on information of previous algorithm runs.
Online Algorithm Selection

**Problem:** Assumption of basic ML is that trainings distributions representative of test distribution.

**The Truth:** Concept drifts are quite likely, i.e., training is not representative of test.

**Idea:** Each solved instance corresponds to new knowledge and the model can be updated.

**Challenge:** We need exploration to update our models.
Online Algorithm Selection as a Bandit Problem
[Degroote et al. 2016]

Idea:

- Each algorithm is a bandit.
- For each instance, we have to decide whether
  1. to exploit our current belief (model 🤔)
  2. To explore algorithms
- Can be modelled with upper confidence bounds (UCB)
- Greedy policy is a surprisingly strong baseline [Degroote et al. 2016]
End-to-End Algorithm Selection

Insight: Most important part is the design of instance features.

Idea: Replace hand-designed features by a neural network
**Idea:** Replace hand-designed features by a neural network

---

**Deep Learning for Algorithm Selection** [Loreggia et al. 2016]

Instance

<table>
<thead>
<tr>
<th>Input layer</th>
<th>32 conv. 3x3 Max pool 2x2 Dropout 0.1</th>
</tr>
</thead>
<tbody>
<tr>
<td>128x128</td>
<td></td>
</tr>
</tbody>
</table>

Convert into image

<table>
<thead>
<tr>
<th>64 conv. 2x2 Max pool 2x2 Dropout 0.2</th>
</tr>
</thead>
<tbody>
<tr>
<td>128 conv. 2x2 Max pool 2x2 Dropout 0.3</td>
</tr>
</tbody>
</table>

Fully connected 1000 nodes Dropout 0.5

Output layer N solvers

Solve instance
Instances as Images

1. Each character translated into ASCII
   - ASCII can be seen as grayscale encoding

2. For \( n \) characters in file, reshape
   into square root\((n)\) x square root\((n)\)

3. Compress into 128x128 pixels

4. Use CNN to classify image

→ works fairly well, but worse than expert features

SAT instance as image
Open Challenges in Algorithm Selection
Open Challenges

1. Generic way for **generating high-quality features**
   - For some domains, we still don’t know good features
   - Deep learning for instance features is still not mature

2. Efficient use of **life-long learning**?
   - Greedy online algorithm selection can’t be the final answer

3. Algorithm selection for **multi-core systems**
   - How to balance exploitation and exploration if we can select $k$ out of $n$ algorithms?
Thank you!