AutoML 101

Marius Lindauer

@LindauerMarius
@AutoML_org

* Slides available at automl.org/talks
Why AutoML?
Success of Machine Learning

- Astronomy
- Energy
- Image Recognition
- Manufacturing
- Maintenance Prediction
- Robotic
- Game Play
- Weather Prediction
- Service
- Traffic Prediction
- Search
- Creative Arts
- Product Recommendation
- Retail
- Media
- Social Media
- Teaching
- Material Design
- Chemistry
- Physics
- Drug Discovery
- Health Care
- Financial Services
- Credit Assignment
- Summary Generation
- Game Play
- Creative Arts
- Product Recommendation
- Retail
- Media
- Social Media
- Teaching
- Material Design
- Chemistry
- Physics
- Drug Discovery
- Health Care
- Financial Services
- Credit Assignment
- Summary Generation
Machine Learning Pipeline

Identify Task → Data Collection → Data cleaning → Feature Engineering → Model Training → Post-Processing → Deployment

Iterative Manual Tuning

Machine Learning Pipeline
Preprocessing?

- Standardization
- Feature Selection
- Outlier Removal
- Missing Feature Imputation
- Embeddings
- Feature Reduction
- PCA
- Kernel PCA
- ICA
- LDA
- NMF
- Truncated SVD

→ We might want more than 1 data preprocessor!
Complexity of Choosing the Preprocessing

- **Naive Assumptions:**
  - only 3 decisions at each level
  - **Possible options:** $3 \times 3 \times 3 = 27$

- **More realistic assumption:**
  - at least 10 decisions at each level
  - **Possible options:** $10 \times 10 \times 10 = 1000$

- Choose 3 preprocessors instead of 1
  - $1000 \times 1000 \times 1000 = 1 000 000 000$

- Still naive!
  - Hyperparameters are often continuous and not discrete
  - **infinite amount of settings!**
There are more than 100 classification algorithms!

Each of these has 2-50 hyperparameters.
Challenges in Designing ML Pipelines

- Identify Task
- Data Collection
- Data cleaning
- Feature Engineering
- Model Training
- Post-Processing
- Deployment

Complex Search Space
Black-Box Problem
Expensive Evaluations
Noise on observations
From Manual ML to Automated ML

Machine Learning Pipeline

Iterative Manual Tuning

Data cleaning → Feature Engineering → Model Training → Post-Processing

Identify Task → Data Collection → Deployment

AutoML

Identify Task → Data Collection → Deployment
Design Decisions taken care by AutoML

### Algorithms

<table>
<thead>
<tr>
<th>classifier</th>
<th>#A</th>
</tr>
</thead>
<tbody>
<tr>
<td>AdaBoost (AB)</td>
<td>4</td>
</tr>
<tr>
<td>Bernoulli naïve Bayes</td>
<td>2</td>
</tr>
<tr>
<td>decision tree (DT)</td>
<td>4</td>
</tr>
<tr>
<td>extreml. rand. trees</td>
<td>5</td>
</tr>
<tr>
<td>Gaussian naïve Bayes</td>
<td>6</td>
</tr>
<tr>
<td>gradient boosting (GB)</td>
<td>3</td>
</tr>
<tr>
<td>kNN</td>
<td>3</td>
</tr>
<tr>
<td>LDA</td>
<td>4</td>
</tr>
<tr>
<td>linear SVM</td>
<td>4</td>
</tr>
<tr>
<td>kernel SVM</td>
<td>7</td>
</tr>
<tr>
<td>multinomial naïve Bayes</td>
<td>2</td>
</tr>
<tr>
<td>passive aggressive</td>
<td>3</td>
</tr>
<tr>
<td>QDA</td>
<td>2</td>
</tr>
<tr>
<td>random forest (RF)</td>
<td>5</td>
</tr>
<tr>
<td>Linear Class. (SGD)</td>
<td>10</td>
</tr>
</tbody>
</table>

### Architecture Design

### Pre-processing

<table>
<thead>
<tr>
<th>preprocessor</th>
<th>#A</th>
</tr>
</thead>
<tbody>
<tr>
<td>extreml. rand. trees prepr.</td>
<td>5</td>
</tr>
<tr>
<td>fast ICA</td>
<td>4</td>
</tr>
<tr>
<td>feature agglomeration</td>
<td>4</td>
</tr>
<tr>
<td>kernel PCA</td>
<td>5</td>
</tr>
<tr>
<td>rand. kitchen sinks</td>
<td>2</td>
</tr>
<tr>
<td>linear SVM prepr.</td>
<td>3</td>
</tr>
<tr>
<td>no preprocessing</td>
<td>-</td>
</tr>
<tr>
<td>rystroem sampler</td>
<td>5</td>
</tr>
<tr>
<td>PCA</td>
<td>2</td>
</tr>
<tr>
<td>polynomial</td>
<td>3</td>
</tr>
<tr>
<td>random trees embed.</td>
<td>4</td>
</tr>
<tr>
<td>select percentile</td>
<td>2</td>
</tr>
<tr>
<td>select rates</td>
<td>3</td>
</tr>
<tr>
<td>one-hot encoding</td>
<td>2</td>
</tr>
<tr>
<td>imputation</td>
<td>1</td>
</tr>
<tr>
<td>balancing</td>
<td>1</td>
</tr>
<tr>
<td>rescaling</td>
<td>1</td>
</tr>
</tbody>
</table>
Importance of Design Decisions in ML
(example on one specific dataset)

Choosing the correct algorithm → 17% improv.

Optimized hyperparameters → 20% - 29% improvement
## Benefits of AutoX Methods

<table>
<thead>
<tr>
<th>Domain</th>
<th>Default vs. Optimized</th>
</tr>
</thead>
<tbody>
<tr>
<td>Answer Set Solving [Gebser et al. 2011]</td>
<td>up to 14× speedup</td>
</tr>
<tr>
<td>AI Planning [Vallati et al. 2013]</td>
<td>up to 40× speedup</td>
</tr>
<tr>
<td>Mixed Integer Programming [Hutter et al. 2010]</td>
<td>up to 52× speedup</td>
</tr>
<tr>
<td>Satisfiability Solving [Hutter et al. 2017]</td>
<td>up to 3000× speedup</td>
</tr>
<tr>
<td>Minimum Vertex Cover [Wagner et al. 2017]</td>
<td>up to 9% absolute impr.</td>
</tr>
<tr>
<td>Machine Learning [Feuer et al. 2015]</td>
<td>up to 35% absolute impr.</td>
</tr>
<tr>
<td>Deep Learning [Zimmer et al. 2020]</td>
<td>up to 49% absolute impr.</td>
</tr>
</tbody>
</table>
Success Stories in AutoML
International Challenge on AutoML I + II

Problem Setting

Constraints

- Time budget (only minutes)
- Memory constraint (few GB)
- Compute power (few CPUs)

Remarks

- More than 100 teams in first challenge
- 44 teams in second challenge
- Both AutoML and human teams

Results

Auto-Sklearn
[Feurer et al. 2015, 2018, 2020]
Shape Error Prediction in Milling Processes
[Denkena et al. 2020]

Problem Setting

Image Source: Dittrich et al. 2018

Search Space

Remarks

- Application of AutoML out-of-box
- Better results than Phd student of machining after spending substantial time
- Reading in the data format cost the most dev. time to let AutoML run

Results

State of the art
AutoML
Main Ideas
AutoML

Optimization and automation of tedious design decisions of a complete ML pipeline in order to obtain a model with peak performance.

**Bayesian Optimization (BO)**
Estimate the model's performance for unknown hyperparameters.

**Evolutionary Algorithm (EA)**
Select, mutate and recombine configurations.

**Hyperparameter Optimization (HPO)**
Search for the best hyperparameter configuration given an algorithm.

**Speed Up**
Meta-learning across datasets, Grey-box optimization/learning curve prediction, Multi-fidelity optimization

**Neural Architecture Search (NAS)**
Search for the best neural network architecture on different hierarchical levels given a task.

**Meta-Learning**
Learn across tasks.

**Algorithm Selection**
Given a task choose the best algorithm based on performance perfection.

**Algorithm Portfolio**

**Configuration Space**
Generic ML Algorithms, Random Forest, Linear Regression, Neural Networks

**Pre-Processing**

DATA

GAUSSIAN PROCESS

RANDOM FOREST

JOIN GATE MODEL

ACQUISITION FUNCTION

TRAIN AND EVALUATE

CLASSIC ML

TRAIN model

MULTI-CRITERIA OPTIMIZATION

SINGLE-CRITERIA OPTIMIZATION

EVALUATE

MODEL(s)

INTERPRETE

START CONFIGURATION

DYNAMIC ALGORITHM CONFIGURATION (DAC)

LEARNING TO LEARN

POPULATION-BASED TRAINING

TRAIN model

CC BY-SA 4.0 from AI-Campus.org and AutoML.org
AutoML 101

Optimization and automation of tedious design decisions of a complete ML pipeline in order to obtain a model with peak performance.

**AutoML**

**Bayesian Optimization (BO)**
Estimate the model's performance for unknown hyperparameters.

**Evolutionary Algorithm (EA)**
Select, mutate and recombine configurations.

**Hyperparameter Optimization (HPO)**
Search for the best hyperparameter configuration given an algorithm.

**Classic ML**
Train model

**Single-Criteria Optimization**
Train model

**Multi-Criteria Optimization**

**Speed Up**
- Meta-learning across datasets
- Grey-box optimization/learning curve prediction
- Multi-fidelity optimization

**Neural Architecture Search (NAS)**
Search for the best neural network architecture on different hierarchical levels given a task.

**Meta-Learning**
Learn across tasks.

**Learning to Learn**

**Population-Based Training**

**Dynamic Algorithm Configuration (DAC)**

**Interpretation**
Model(s)

**Data**

**Configuration Space**
- Generic ML Algorithms
  - Random Forest
  - Linear Regression
- Neural Networks

**Algorithm Selection**
Given a task choose the best algorithm based on performance perfection.

**Algorithm Portfolio**
AutoML Optimizer

Bayesian Optimization

- Global optimization strategy
- Very sample efficient
- Very efficient for small/med. config. spaces

Evolutionary Algorithms

- Population based-approach
- Strong performance for longer budget
- Easy to parallelize

Reinforcement Learning

- Learning of a policy
- Can learn a generalizable policy
- Human-like approach
**AutoML**

Optimization and automation of tedious design decisions of a complete ML pipeline in order to obtain a model with peak performance.

**Hyperparameter Optimization (HPO)**

Search for the best hyperparameter configuration given an algorithm.

**Bayesian Optimization (BO)**

Estimate the model's performance for unknown hyperparameters.

**Evolutionary Algorithm (EA)**

Select, mutate and recombine configurations.

**Configuration Space**

- Generic ML Algorithms
  - Random Forest
  - Linear Regression
- Neural Networks

**Algorithm Selection**

Given a task choose the best algorithm based on performance perfection.

**Algorithm Portfolio**

**Speed Up**

Meta-learning across datasets

**Neural Architecture Search (NAS)**

Search for the best neural network architecture on different hierarchical levels given a task.

**Dynamic Algorithm Configuration (DAC)**

**Meta-Learning**

Learn across tasks.

**Population-Based Training**

**Hyperparameter Optimization (HPO)**

Pick next hyperparameters

**Classic ML**

Train model

**Evaluate model parameters**

**Single-Criteria Optimization**

**Multi-Criteria Optimization**

**Interpret**

Model(s)

**Evaluate**

CC BY-SA 4.0 from AI-Campus.org and AutoML.org
AutoML: Model Selection

Hold-Out

| Training | Validation |

Cross Validation

| Fold 1 | Fold 2 | Fold 3 | Fold 4 |

Error vs. Epochs

- Full Evaluation
- Successive Halving
  
  [Jamieson & Talwaker 2015, Li et al. 2018]
AutoML

Optimization and automation of tedious design decisions of a complete ML pipeline in order to obtain a model with peak performance.

**Configuration Space**
- Generic ML Algorithms
  - Random Forest
  - Linear Regression
- Neural Networks
- PRE-PROCESSING

**Algorithm Selection**
Given a task choose the best algorithm based on performance perfection.

**Algorithm Portfolio**

**Bayesian Optimization (BO)**
Estimate the model's performance for unknown hyperparameters.

**Evolutionary Algorithm (EA)**
Select, mutate and recombine configurations.

**Hyperparameter Optimization (HPO)**
Search for the best hyperparameter configuration given an algorithm.

**Single-Criteria Optimization**
- Classic ML
  - Train model
  - Evaluate model parameters

**Multi-Criteria Optimization**

**Speed Up**
- Meta-learning across datasets
- Grey-box optimization/learning curve prediction
- Multi-fidelity optimization

**Neural Architecture Search (NAS)**
Search for the best neural network architecture on different hierarchical levels given a task.

**Meta-Learning**
Learn across tasks.

**Learning to Learn**
Population-based training

**Dynamic Algorithm Configuration (DAC)**

**Interpretation**

**Model(s)**

CC BY-SA 4.0 from AI-Campus.org and AutoML.org
Warmstarting via Meta-Learning

[Feurer et al. 2015, Lindauer & Hutter. 2017]
Warmstarting via Meta-Learning  

[Lindauer & Hutter. 2017]
AutoML

Optimization and automation of tedious design decisions of a complete ML pipeline in order to obtain a model with peak performance.

- **Gaussian Process**: Estimate the model's performance for unknown hyperparameters.
- **Bayesian Optimization (BO)**: Select, mutate and recombine configurations.
- **Evolutionary Algorithm (EA)**: Search for the best hyperparameter configuration given an algorithm.

**Hyperparameter Optimization (HPO)**

- **Classic ML**: Search for hyperparameter configuration.
- **Single-Criteria Optimization**: Meta-learning across datasets.
- **Multi-Criteria Optimization**: Grey-box optimization/learning curve prediction.
- **Multi-fidelity optimization**: Multi-fidelity optimization.

**Neural Architecture Search (NAS)**

Search for the best neural network architecture on different hierarchical levels given a task.

**Meta-Learning**

Learn across tasks.

**Algorithm Selection**

Given a task choose the best algorithm based on performance perfection.

**Algorithm Portfolio**

- **Configuration Space**
  - Generic ML Algorithms
  - Random Forest
  - Linear Regression

- **ML Algorithms**
  - Neural Networks

**Pre-Processing**

**Data**

**Interpret**
AutoML Ensembles

- If ensemble members make uncorrelated errors
- Already a diverse set of ensemble members can perform well
- AutoML can help to find even better ensembles
Open Source Software Projects
Auto-Sklearn [Feurer et al. 2015, 2018, 2020]

Takes care of well-performing ML-pipeline

- Winner of 1st and 2nd AutoML Challenge
- Improved efficiency in Version 2.0 by
  - Meta-learning, multi-fidelity optimization, automating AutoML

Easy-to-use

```python
import autosklearn.classification as cls
automl = cls.AutoSklearnClassifier()
automl.fit(X_train, y_train)
y_hat = automl.predict(X_test)
```
Auto-PyTorch [Mendoza et al. 2019, Zimmer et al. 2021]

```python
autoPyTorch = AutoNetClassification("tiny_cs",  # config preset
               log_level=\'info\',
               max_runtime=300,
               min_budget=30,
               max_budget=90)

autoPyTorch.fit(X_train, y_train, validation_split=0.3)

y_pred = autoPyTorch.predict(X_test)
```

- Strong performance against other state-of-the-AutoML tools on tabular data
- Even competitive on image data against gradient-based methods
- Efficiency due to meta-learning, multi-fidelity optimization and ensembling
**SMAC** [Hutter et al. 2011, Lindauer et al. 2017]

```python
x, cost, _ = fmin_smac(func=b Branin,  # function
    x0=[0, 0],  # default configuration
    bounds=[(-5, 10), (0, 15)],  # limits
    maxfun=10,  # maximum number of evaluations
    rng=3)  # random seed
print("Optimum at {} with cost of ").format(x, cost))
```

- Working horse for Auto-Sklearn
  - Soon also for Auto-PyTorch

- Implements state-of-the-art approaches for
  - Bayesian optimization
  - Multi-fidelity optimization
    - E.g., successive halving, hyperband, BOHB
  - Algorithm configuration
    - Robust configurations across many tasks
Pros and Cons of AutoML
Pros and Cons

+ Saves human developer time  
- Costs compute time

- Costs compute time
- Not always better than experts
- Cannot help you with defining the underlying task or business case
- Cannot (directly) help you with getting more and better data
- Adds another layer of a black-box on top of ML (?)
- Harder to consider expert knowledge (?)
AutoML vs. Expert Knowledge
Use the Gained Time for Feature Engineering

- All of these steps are important
- Often iterative cycle between these
- If one of these gets automated, more time for others available

- Feature engineering is often one of the best places to consider expert knowledge
  - Nevertheless, it can also be automated
Cut Down the Search Space

- SVM
- linear
- Resnet
- FF-Net

Expert Knowledge

- SVM
- linear
- Resnet
- FF-Net
Expert Knowledge as Probability Distributions

[Luis et al. 2020]

→ Increases efficiency of AutoML
What’s next?
AutoML + Meta-Learning

- Amount of data is continuously increasing
  → Models have to be updated or even trained from scratch if a concept drift occurs
  → ML pipeline needs adjustments

- AutoML could help to enable maintainability also in the long run [Celik & Vanschoren 2020]

- Similarly, AutoML can help even if the underlying ML algorithm changes [Stoll et al. 2020]
AutoML: Explainable AutoML
[Biedenkapp et al. 2017, 2018, 2019, Moosbauer et al. 2021]

- Users want to know more than the result
  For example:
  - Which design decisions were important?
  - Why was the returned pipeline chosen?
  - Was the approach of the AutoML tool appropriate for the dataset at hand?
  - ...

- What would be interesting for you?

<table>
<thead>
<tr>
<th></th>
<th>fANOVA</th>
<th>LPI</th>
</tr>
</thead>
<tbody>
<tr>
<td>discount</td>
<td>19.32</td>
<td>38.88</td>
</tr>
<tr>
<td>learning rate</td>
<td>3.70</td>
<td>35.4</td>
</tr>
<tr>
<td>batch size</td>
<td>15.77</td>
<td>21.5</td>
</tr>
<tr>
<td># units 1</td>
<td>1.86</td>
<td>0.07</td>
</tr>
<tr>
<td># units 2</td>
<td>0.39</td>
<td>0.01</td>
</tr>
</tbody>
</table>
Dynamic Algorithm Control via RL


Traditional AutoML: Blackbox-Optimization

Configure once

Future: Reinforcement learning to learn to adjust hyperparameters over time

Algorithm

Dynamic Configurator

Internal Heuristics
Learning more about AutoML?
Further Material

AutoML online course starting April 2021
Our Vision: Democratization of AI

1. We need tools s.t. AI is **easy-to-use**

2. **Efficient development** of new AI applications

3. AutoML will leverage **interdisciplinary applications**: ML + ?

4. **Improved understanding** of AI systems
Thank you!