AutoML: Beyond AutoML

Capping of Runtimes

Bernd Bischl   Frank Hutter   Lars Kotthoff
Marius Lindauer   Joaquin Vanschoren

Adaptive capping [Hutter et al. 2009]

- Assumptions
  - optimization of runtime
  - each configuration run has a time limit (e.g., 300 sec)

Adaptive capping [Hutter et al. 2009]

- Assumptions
  - optimization of runtime
  - each configuration run has a time limit (e.g., 300 sec)
- E.g., $\hat{\boldsymbol{\lambda}}$ needed 1 sec to solve i_1
  - Do we need to run $\boldsymbol{\lambda}'$ for 300 sec?
  - Terminate evaluation of $\boldsymbol{\lambda}'$ once guaranteed to be worse than $\hat{\boldsymbol{\lambda}}$

Adaptive capping [Hutter et al. 2009]

- Assumptions
  - optimization of runtime
  - each configuration run has a time limit (e.g., 300 sec)
- E.g., $\hat{\boldsymbol{\lambda}}$ needed 1 sec to solve i_1
  - Do we need to run $\boldsymbol{\lambda}'$ for 300 sec?
  - Terminate evaluation of $\boldsymbol{\lambda}'$ once guaranteed to be worse than $\hat{\boldsymbol{\lambda}}$

$\rightsquigarrow$ To compare against $\hat{\boldsymbol{\lambda}}$ based on N runs,
we can terminate evaluation of $\boldsymbol{\lambda}'$ after time $\sum_{k=1}^{N} c(\hat{\boldsymbol{\lambda}}, i_k)$

Toy-Example: Adaptive capping

runtime cutoff $\kappa = 300$, comparison based on 2 instances

	i_1	i_2
$\hat{\boldsymbol{\lambda}}$	4	2

Toy-Example: Adaptive capping

runtime cutoff $\kappa = 300$, comparison based on 2 instances

	i_1	i_2
$\hat{\boldsymbol{\lambda}}$	4	2
Without adaptive capping		
$\boldsymbol{\lambda}'$		

Toy-Example: Adaptive capping

runtime cutoff $\kappa = 300$, comparison based on 2 instances

	i_1	i_2
$\hat{\boldsymbol{\lambda}}$	4	2
Without adaptive capping		
$\boldsymbol{\lambda}'$	3	

Toy-Example: Adaptive capping

runtime cutoff $\kappa = 300$, comparison based on 2 instances

	i_1	i_2
$\hat{\boldsymbol{\lambda}}$	4	2
Without adaptive capping		
$\boldsymbol{\lambda}'$	3	300

Toy-Example: Adaptive capping

runtime cutoff $\kappa = 300$, comparison based on 2 instances

	i_1	i_2
$\hat{\boldsymbol{\lambda}}$	4	2
Without adaptive capping		
$\boldsymbol{\lambda}'$	3	300

$\rightarrow$ reject $\boldsymbol{\lambda}'$ (cost: 303)

Toy-Example: Adaptive capping

runtime cutoff $\kappa = 300$, comparison based on 2 instances

	i_1	i_2
$\hat{\boldsymbol{\lambda}}$	4	2
Without adaptive capping		
$\boldsymbol{\lambda}'$	3	300
		$\rightarrow$ reject $\boldsymbol{\lambda}'$ (cost: 303)
With adaptive capping		
$\boldsymbol{\lambda}'$		

Toy-Example: Adaptive capping

runtime cutoff $\kappa = 300$, comparison based on 2 instances

	i_1	i_2
$\hat{\boldsymbol{\lambda}}$	4	2
Without adaptive capping		
$\boldsymbol{\lambda}'$	3	300
		$\rightarrow$ reject $\boldsymbol{\lambda}'$ (cost: 303)
With adaptive capping		
$\boldsymbol{\lambda}'$	3	

Toy-Example: Adaptive capping

runtime cutoff $\kappa = 300$, comparison based on 2 instances

	i_1	i_2
$\hat{\boldsymbol{\lambda}}$	4	2
Without adaptive capping		
$\boldsymbol{\lambda}'$	3	300
		$\rightarrow$ reject $\boldsymbol{\lambda}'$ (cost: 303)
With adaptive capping		
$\boldsymbol{\lambda}'$	3	300

Toy-Example: Adaptive capping

runtime cutoff $\kappa = 300$, comparison based on 2 instances

	i_1	i_2
$\hat{\boldsymbol{\lambda}}$	4	2
Without adaptive capping		
$\boldsymbol{\lambda}'$	3	300
		$\rightarrow$ reject $\boldsymbol{\lambda}'$ (cost: 303)
With adaptive capping		
$\boldsymbol{\lambda}'$	3	300
	$\rightarrow$ cut off after $\kappa = 4$ seconds, reject $\boldsymbol{\lambda}'$ (cost: 7)	

Toy-Example: Adaptive capping

runtime cutoff $\kappa = 300$, comparison based on 2 instances

	i_1	i_2
$\hat{\boldsymbol{\lambda}}$	4	2
Without adaptive capping		
$\boldsymbol{\lambda}'$	3	300
		$\rightarrow$ reject $\boldsymbol{\lambda}'$ (cost: 303)
With adaptive capping		
$\boldsymbol{\lambda}'$	3	300
	$\rightarrow$ cut off after $\kappa = 4$ seconds, reject $\boldsymbol{\lambda}'$ (cost: 7)	

Note: To combine adaptive capping with BO, we need to impute the censored observations caused by adaptive capping. [Hutter et al. 2011]

Overview of Racing and Adaptive Capping

Input : candidate configurations $\mathbf{\Lambda}_{new}$, cutoff κ_{max}, previously evaluated runs $\mathcal{D}_{\text{Hist}}$, budget T, incumbent $\hat{\boldsymbol{\lambda}}$

while $\mathbf{\Lambda}_{new}$ *not empty* **do**

$\boldsymbol{\lambda}^{(t)} \leftarrow$ getNext($\mathbf{\Lambda}_{new}$);

[. . . add new run for incumbent . . .];

while *true* **do**

$\mathcal{I}^{+}, \mathbf{s}^{+} \leftarrow$getAlreadyEvaluatedOn($\hat{\boldsymbol{\lambda}}, \mathcal{D}_{\text{Hist}}$);

$\mathcal{I}^{(t)}, \mathbf{s}^{(t)} \leftarrow$getAlreadyEvaluatedOn($\boldsymbol{\lambda}^{(t)}, \mathcal{D}_{\text{Hist}}$);

$i^{(t)}, s^{(t)} \leftarrow$ drawn uniformly at random from $\mathcal{I}_{+} \setminus \mathcal{I}^{(t)}$ and $\mathbf{s}^{+} \setminus \mathbf{s}^{(\mathbf{t})}$;

$\kappa^{(i)} \leftarrow$ AdaptCutoff($\kappa_{max}, \langle(\boldsymbol{\lambda}^{(j)}, c^{(j)})\rangle_{\boldsymbol{\lambda}^{(j)}=\boldsymbol{\lambda}^{+}}) \cdot \xi$;

$c_i \leftarrow$ EvaluateRun($\boldsymbol{\lambda}^{(t)}, i^{(t)}, s^{(i)}, \kappa^{(i)}$);

$\mathcal{D}_{\text{Hist}} \leftarrow \mathcal{D}_{\text{Hist}} \cup (\boldsymbol{\lambda}^{(t)}, i^{(t)}, s^{(t)}, c^{(t)})$;

if *average cost of* $\boldsymbol{\lambda}^{(t)} >$ *average cost of* $\hat{\boldsymbol{\lambda}}$ *across* $\mathcal{I}^{(t)}$ *and* $\mathbf{s}^{(t)}$ **then**

break;

else if *average cost of* $\boldsymbol{\lambda}^{(t)} <$ *average cost of* $\hat{\boldsymbol{\lambda}}$ *and* $\mathcal{I}^{+} = \mathcal{I}^{(t)}$ *and* $\mathbf{s}^{(t)} = \mathbf{s}^{+}$ **then**

$\hat{\boldsymbol{\lambda}} \leftarrow \boldsymbol{\lambda}^{(t)}$;

if *time spent exceeds* T *or* $\mathbf{\Lambda}_{new}$ *is empty* **then**

return $\hat{\boldsymbol{\lambda}}, \mathcal{D}_{Hist}$