Performance is not Boolean

Supporting Scalar Configuration Variables in NFP Models

Daniel Friesel, Olaf Spinczyk
daniel.friesel@uos.de

March 18, 2022

Universität Osnabrück / Arbeitsgruppe Eingebettete Softwaresysteme

Motivation

```
                        .config - Configuration
 System Logging Utilities
    [*] klogd (5.7 kb)
          *** klogd should not be used together with syslog to kernel printk buffer ***
    [*]   Use the klogctl() interface
    [*] logger (6.3 kb)
    [*] logread (4.8 kb)
    [*]   Double buffering
    [*] syslogd (13 kb)
    [*]   Rotate message files
    [*]   Remote Log support
    [*]   Support -D (drop dups) option
    [*]   Support syslog.conf
    [ ]   Include milliseconds in timestamps
        (256) Read buffer size in bytes
    [*]   Circular Buffer support
        (16)    Circular buffer size in Kbytes (minimum 4KB)

F1Help F2SymInfo F3Help 2 F4ShowAll F5Back F6Save F7Load F8SymSearch F9Exit
```

Motivation

```
                         .config - Configuration
 System Logging Utilities
    [*] klogd (5.7 kb)
          *** klogd should not be used together with syslog to kernel printk buffer ***
    [*]   Use the klogctl() interface
    [*] logger (6.3 kb)
    [*] logread (4.8 kb)
    [*]   Double buffering
    [*] syslogd (13 kb)
    [*]   Rotate message files
    [*]   Remote Log support
    [*]   Support -D (drop dups) option
    [*]   Support syslog.conf
    [ ]   Include milliseconds in timestamps
        (256) Read buffer size in bytes
    [*]   Circular Buffer support
        (16)    Circular buffer size in Kbytes (minimum 4KB)

F1Help F2SymInfo F3Help 2 F4ShowAll F5Back F6Save F7Load F8SymSearch F9Exit
```

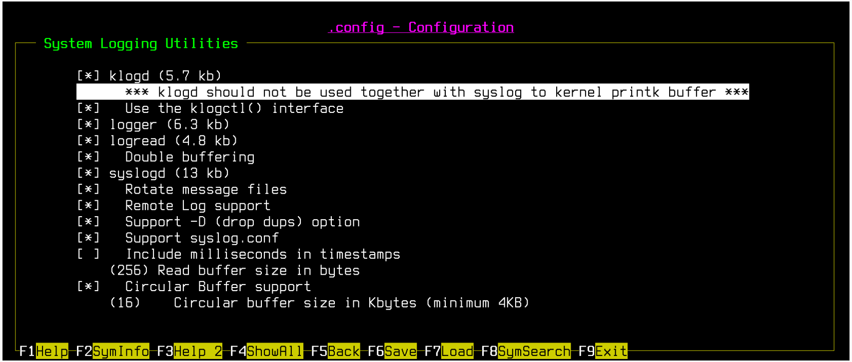
Motivation

```
                          .config - Configuration
 System Logging Utilities
    [*] klogd (5.7 kb)
          *** klogd should not be used together with syslog to kernel printk buffer ***
    [*]   Use the klogctl() interface
    [*] logger (6.3 kb)
    [*] logread (4.8 kb)
    [*]   Double buffering
    [*] syslogd (13 kb)
    [*]   Rotate message files
    [*]   Remote Log support
    [*]   Support -D (drop dups) option
    [*]   Support syslog.conf
    [ ]   Include milliseconds in timestamps
        (256) Read buffer size in bytes
    [*]   Circular Buffer support
        (16)    Circular buffer size in Kbytes (minimum 4KB)

F1Help F2SymInfo F3Help 2 F4ShowAll F5Back F6Save F7Load F8SymSearch F9Exit
```

Motivation

```
                         .config - Configuration
 System Logging Utilities
        [*] klogd (5.7 kb)
              *** klogd should not be used together with syslog to kernel printk buffer ***
        [*]   Use the klogctl() interface
        [*] logger (6.3 kb)
        [*] logread (4.8 kb)
        [*]   Double buffering
        [*] syslogd (13 kb)
        [*]   Rotate message files
        [*]   Remote Log support
        [*]   Support -D (drop dups) option
        [*]   Support syslog.conf
        [ ]   Include milliseconds in timestamps
            (256) Read buffer size in bytes
        [*]   Circular Buffer support
            (16)    Circular buffer size in Kbytes (minimum 4KB)

 F1Help F2SymInfo F3Help 2 F4ShowAll F5Back F6Save F7Load F8SymSearch F9Exit
```

Motivation



- Automatic generation of detailed and accurate models

→ predict non-functional properties (NFPs) of system / workload from boolean and scalar features

NFP Models

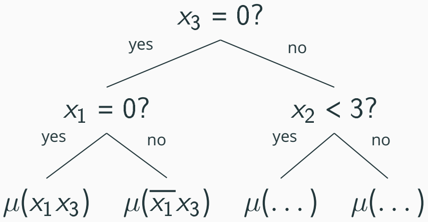
$$\underset{\vec{x} \in M}{\text{Configuration}} \longrightarrow \boxed{\underset{f}{\text{NFP model}}} \longrightarrow \underset{y \in \mathbb{R}}{\text{RAM / Latency / } \ldots}$$

NFP Models

Configuration $\vec{x} \in \mathbb{R}^n$ → NFP model f → RAM / Latency / . . . $y \in \mathbb{R}$

Two common approaches:

- Classification and Regression Trees (CART) [Guo+18; Nai+20; Jam+18]



NFP Models

$$\underset{\vec{x} \in \mathbb{R}^n}{\text{Configuration}} \longrightarrow \boxed{\underset{f}{\text{NFP model}}} \longrightarrow \underset{y \in \mathbb{R}}{\text{RAM / Latency / ...}}$$

Two common approaches:

- Classification and Regression Trees (CART) [Guo+18; Nai+20; Jam+18]
- Linear Regression [Sie+15]

$$\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_1 x_2 + \ldots$$

NFP Models

Configuration $\vec{x} \in \{0, 1\}^n$ $\longrightarrow$ NFP model f $\longrightarrow$ RAM / Latency / ... $y \in \mathbb{R}$

Two common approaches:

- Classification and Regression Trees (CART) [Guo+18; Nai+20; Jam+18]
- Linear Regression [Sie+15]

But:

- Often limited to boolean features (→ data-efficient sampling) [Per+21]

NFP Models

Configuration $\vec{x} \in \{\mathbb{R} \cup \{\perp\}\}^n$ $\longrightarrow$ NFP model f $\longrightarrow$ RAM / Latency / ... $y \in \mathbb{R}$

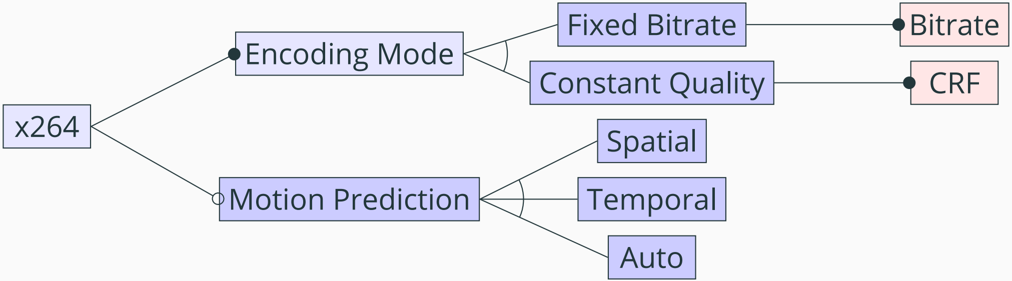
Two common approaches:

- Classification and Regression Trees (CART) [Guo+18; Nai+20; Jam+18]
- Linear Regression [Sie+15]

But:

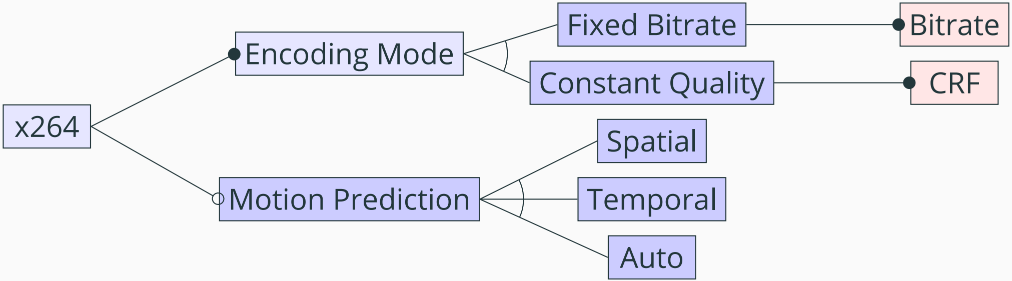
- Often limited to boolean features (→ data-efficient sampling) [Per+21]
- Non-linear effects
- Features may be undefined

Feature Extraction



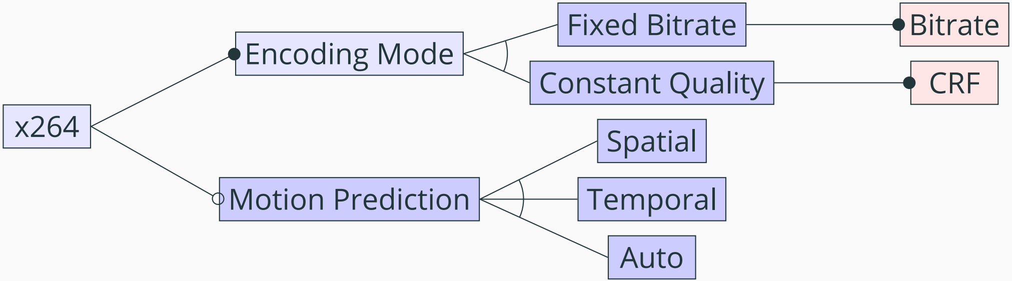
- Configuration / Workload $\xrightarrow{?}$ feature vector $\vec{x}$

Feature Extraction



- Configuration / Workload $\overset{?}{\rightarrow}$ feature vector $\vec{x}$
- $x_{\text{fb}}, x_{\text{cq}} \in \{0,1\}, \sum_{x_i} = 1$; $\quad x_{\text{spatial}}, x_{\text{temporal}}, x_{\text{auto}} \in \{0,1\}, \sum_{x_i} \leq 1$
- $x_{\text{bitrate}}, x_{\text{crf}} \in \mathbb{R}$

Feature Extraction



- Configuration / Workload $\overset{?}{\rightarrow}$ feature vector $\vec{x}$
- $x_{\text{fb}}, x_{\text{cq}} \in \{0, 1\}, \sum_{x_i} = 1$; $\quad x_{\text{spatial}}, x_{\text{temporal}}, x_{\text{auto}} \in \{0, 1\}, \sum_{x_i} \leq 1$
- $x_{\text{bitrate}}, x_{\text{crf}} \in \{\mathbb{R} \cup \{\bot\}\}$
- Constant Quality mode selected $\rightarrow x_{\text{bitrate}} = \bot$

Regression Model Trees (RMT)

- Regression Trees
  - ✓ tree generation can be adapted for undefined variables
  - ! piecewise constant approximation of complex scalar functions

Regression Model Trees (RMT)

- Regression Trees
  - ✓ tree generation can be adapted for undefined variables
  - ! piecewise constant approximation of complex scalar functions
- (Linear) Regression Models
  - ✓ accurate expression of (linear) functions
  - ! complex, $\vec{x} \in \mathbb{R}^n$

Regression Model Trees (RMT)

- Regression Trees
  - ✓ tree generation can be adapted for undefined variables
  - ! piecewise constant approximation of complex scalar functions
- (Linear) Regression Models
  - ✓ accurate expression of (linear) functions
  - ! complex, $\vec{x} \in \mathbb{R}^n$

→ Regression Model Trees

$\vdots$

$x_{\text{fb}} = 0?$

yes / no

boolean-only

$\beta_0 + \beta_1 \log(x_{\text{crf}})$ $\quad$ $\beta'_0 + \frac{\beta'_1}{x_{\text{bitrate}}}$

scalar-only

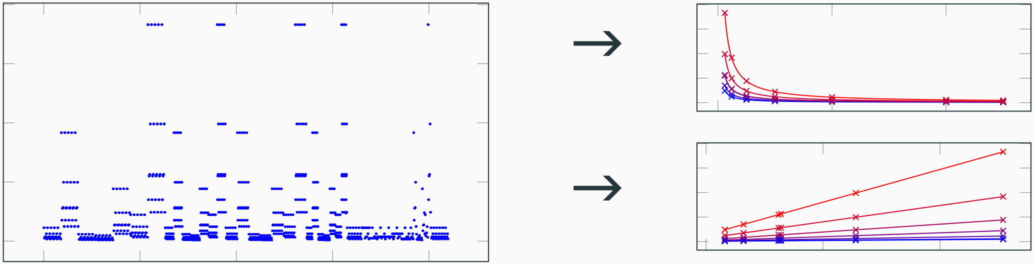
RMT Generation

Input: observations $S = \{y_1, y_2, \ldots\}$, configurations $\vec{x}_1, \vec{x}_2, \ldots$

RMT Generation

Input: observations $S = \{y_1, y_2, \dots\}$, configurations $\vec{x}_1, \vec{x}_2, \dots$

1. stop criterion? → build function for S (ignore boolean features) [FBS18]



RMT Generation

Input: observations $S = \{y_1, y_2, \ldots\}$, configurations $\vec{x}_1, \vec{x}_2, \ldots$

1. stop criterion? → build function for S (ignore boolean features) [FBS18]
2. for each boolean x_i : $\overline{S_i} =_{df} \{y_k | x_k = 0\}$; $S_i =_{df} \{y_k | x_k = 1\}$ [Bre+84]

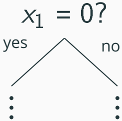
	y_1	y_2	y_3	y_4
x_1	0	0	1	1
x_2	0	1	0	1
	$\overline{S_1}$	$\overline{S_1}$	S_1	S_1
	$\overline{S_2}$	S_2	$\overline{S_2}$	S_2

RMT Generation

Input: observations $S = \{y_1, y_2, \ldots\}$, configurations $\vec{x}_1, \vec{x}_2, \ldots$

1. stop criterion? → build function for S (ignore boolean features) [FBS18]
2. for each boolean x_i : $\overline{S_i} =_{df} \{y_k | x_k = 0\}$; $S_i =_{df} \{y_k | x_k = 1\}$ [Bre+84]
3. Select x_i with lowest loss → RMT node "$x_i = 0?$"

	y_1	y_2	y_3	y_4
x_1	0	0	1	1
x_2	0	1	0	1
	$\overline{S_1}$	$\overline{S_1}$	S_1	S_1
	$\overline{S_2}$	S_2	$\overline{S_2}$	S_2

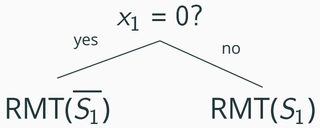


$$\mathrm{SSR}(i) =_{df} \sum_{y \in S_i} (\mu(S_i) - y)^2 + \sum_{y \in \overline{S_i}} (\mu(\overline{S_i}) - y)^2$$

RMT Generation

Input: observations $S = \{y_1, y_2, \ldots\}$, configurations $\vec{x}_1, \vec{x}_2, \ldots$

1. stop criterion? → build function for S (ignore boolean features) [FBS18]
2. for each boolean x_i : $\overline{S_i} =_{df} \{y_k | x_k = 0\}$; $S_i =_{df} \{y_k | x_k = 1\}$ [Bre+84]
3. Select x_i with lowest loss → RMT node "$x_i = 0$?"
4. Recursion → $S := \overline{S_i}$ / $S := S_i$



RMT Generation

Input: observations $S = \{y_1, y_2, \ldots\}$, configurations $\vec{x}_1, \vec{x}_2, \ldots$

1. stop criterion? → build function for S (ignore boolean features) [FBS18]
2. for each boolean x_i : $\overline{S_i} =_{df} \{y_k | x_k = 0\}$; $S_i =_{df} \{y_k | x_k = 1\}$ [Bre+84]
3. Select x_i with lowest loss → RMT node "$x_i = 0$?"
4. Recursion → $S := \overline{S_i}$ / $S := S_i$

Claim: RMT are more accurate and do not increase model size

Evaluation Setup

- **Busybox** multi-call binary: ROM (.text), static RAM (.data, .bss)
- **Kratos** embedded research OS: ROM, static RAM
- **Multipass** embedded research OS: ROM, static RAM
- **x264** video encoding: duration, output file size
- 13 to $1,018$ features; 2 to $30\,\%$ scalar

Evaluation Setup

- **Busybox** multi-call binary: ROM (.text), static RAM (.data, .bss)
- **Kratos** embedded research OS: ROM, static RAM
- **Multipass** embedded research OS: ROM, static RAM
- **x264** video encoding: duration, output file size
- 13 to $1,018$ features; 2 to $30\,\%$ scalar

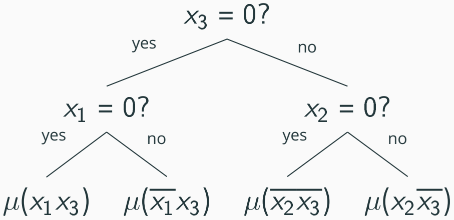
Benchmarks:

- Busybox: random sampling
- Others: random sampling with neighbourhood exploration
- Data-efficient sampling is out of scope

NFP Model Lineup

Bool:

- DECART (data-efficient classification and regression trees) [Guo+18]



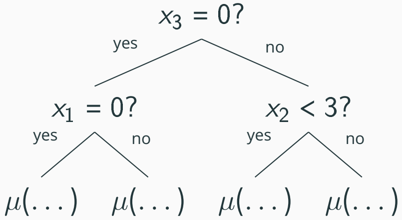
NFP Model Lineup

Bool:

- DECART (data-efficient classification and regression trees) [Guo+18]

Scalar:

- CART (classification and regression trees) [Bre+84]



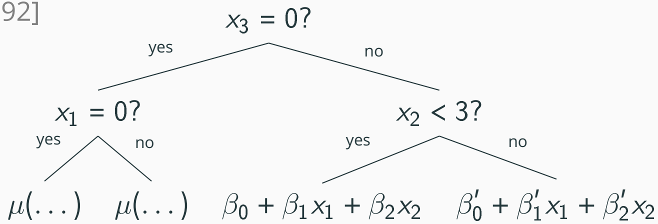
NFP Model Lineup

Bool:

- DECART (data-efficient classification and regression trees) [Guo+18]

Scalar:

- CART (classification and regression trees) [Bre+84]
- LMT (linear model trees) [Qui+92]



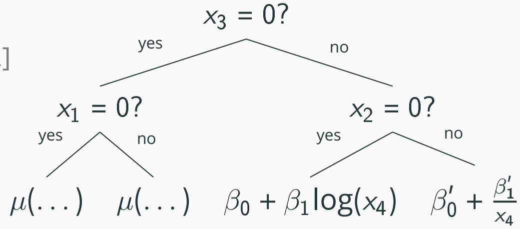
NFP Model Lineup

Bool:

- DECART (data-efficient classification and regression trees) [Guo+18]

Scalar:

- CART (classification and regression trees) [Bre+84]
- LMT (linear model trees) [Qui+92]
- RMT (regression model trees) [FS22]



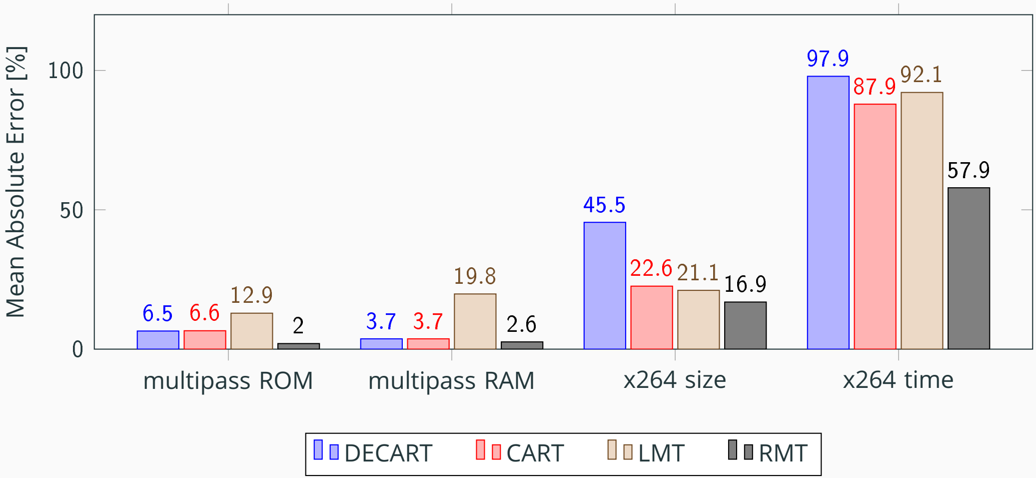
Model Accuracy

Equal performance (deviation < 0.1) for

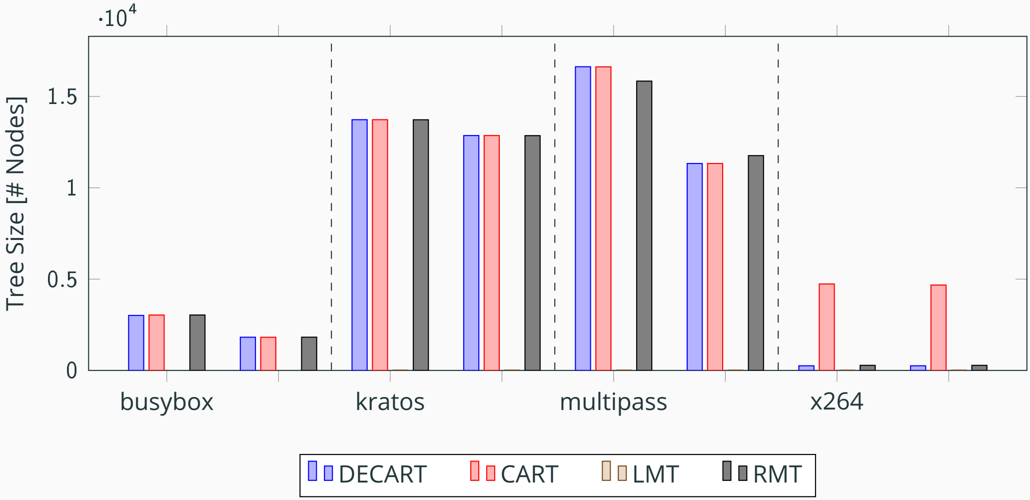
- Busybox ROM/RAM ($0.2\,\%$)
- Kratos ROM ($0.5\,\%$) / RAM ($0.8\,\%$)

→ RMT as accurate as (DE)CART for scalar-independent applications

Model Accuracy



Model Complexity



Conclusion

- Scalar parameters affect non-functional properties
- Scalar-aware NFP models ...
  - have equal or up to $3\times$ lower model error
  - vary in complexity, larger $\not\Rightarrow$ better

Conclusion

- Scalar parameters affect non-functional properties
- Scalar-aware NFP models ...
    - have equal or up to $3\times$ lower model error
    - vary in complexity, larger $\not\Rightarrow$ better
- Regression Model Trees (RMT) achieve lowest model error
- RMT model size within $10\,\%$ of DECART

Conclusion

- Scalar parameters affect non-functional properties
- Scalar-aware NFP models …
  - have equal or up to $3\times$ lower model error
  - vary in complexity, larger $\not\Rightarrow$ better
- Regression Model Trees (RMT) achieve lowest model error
- RMT model size within $10\,\%$ of DECART

WiP:

- Improve RMT generation (more compact $\rightarrow$ explainable models)
- Apply existing approaches for efficient sampling

Conclusion

Goal: NFP visualization and NFP-aware partial auto-configuration

Pareto-optimal configurations ⓘ

- **Cost = 60 € / Throughput = 21.6 FPS** (Task = Segmentation (bounding box) Hardware Platform = Raspberry Pi 4 B (aarch64) NN Architecture = mobilenetv3small)
- **Cost = 480 € / Throughput = 247.0 FPS** (Task = Segmentation (bounding box) NN Architecture = efficientdet_lite0_feature-vector_1 Hardware Platform = Jetson Xavier NX Batch Size = 32)
- **Cost = 117 € / Throughput = 38.0 FPS** (Task = Segmentation (bounding box) Hardware Platform = Jetson Nano NN Architecture = mobilenetv3small Batch Size = 32)
- **Cost = 160 € / Throughput = 243.0 FPS** (Task = Segmentation (bounding box) NN Architecture = mobilenetv2 Hardware Platform = Coral EdgeTPU Dev Board TFLite Optimizations = Int8 Quantization with Sample Data and EdgeTPU Offloading Batch Size = 32)

Batch Size (NEW) 1
Hardware Platform (NEW)
NN Framework (NEW)
Task
NN Architecture (NEW)
TFLite Optimizations (NEW)

task

Classification ☐ 0 € ≈+4.7 FPS
Segmentation (bounding box) ☑
Segmentation (pixel-accurate) ☐ 0 € ≈+4.7 FPS

References i

[Bre+84] Leo Breiman et al. Classification and regression trees. Routledge, 1984. DOI: 10.1201/9781315139470.

[FBS18] Daniel Friesel, Markus Buschhoff, and Olaf Spinczyk. “Parameter-Aware Energy Models for Embedded-System Peripherals”. In: 2018 IEEE 13th International Symposium on Industrial Embedded Systems (SIES). June 2018, pp. 1–4. DOI: 10.1109/SIES.2018.8442096. URL: https://ieeexplore.ieee.org/document/8442096.

[FS22] Daniel Friesel and Olaf Spinczyk. “Performance is not Boolean: Supporting Scalar Configuration Variables in NFP Models”. In: Tagungsband des FG-BS Frühjahrstreffens 2022. Bonn: Gesellschaft für Informatik e.V., 2022. DOI: 10.18420/fgbs2022f-03.

[Guo+18] Jianmei Guo et al. “Data-Efficient Performance Learning for Configurable Systems”. In: Empirical Softw. Engg. 23.3 (June 2018), pp. 1826–1867. ISSN: 1382-3256. DOI: 10.1007/s10664-017-9573-6. URL: https://doi.org/10.1007/s10664-017-9573-6.

References iii

[Jam+18] Pooyan Jamshidi et al. "Learning to Sample: Exploiting Similarities across Environments to Learn Performance Models for Configurable Systems". In: *Proceedings of the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*. ESEC/FSE 2018. Lake Buena Vista, FL, USA: Association for Computing Machinery, 2018, pp. 71–82. ISBN: 9781450355735. DOI: `10.1145/3236024.3236074`. URL: `https://doi.org/10.1145/3236024.3236074`.

[Nai+20] Vivek Nair et al. "Finding Faster Configurations Using FLASH". In: *IEEE Transactions on Software Engineering* 46.7 (2020), pp. 794–811. DOI: `10.1109/TSE.2018.2870895`.

[Per+21] Juliana Alves Pereira et al. “Learning software configuration spaces: A systematic literature review”. In: *Journal of Systems and Software* 182 (2021), p. 111044. ISSN: 0164-1212. DOI: https://doi.org/10.1016/j.jss.2021.111044. URL: https://www.sciencedirect.com/science/article/pii/S0164121221001412.

[Qui+92] John R Quinlan et al. “Learning with continuous classes”. In: *5th Australian joint conference on artificial intelligence*. Vol. 92. World Scientific. 1992, pp. 343–348.

References v

[Sie+15] Norbert Siegmund et al. “Performance-Influence Models for Highly Configurable Systems”. In: Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering. ESEC/FSE 2015. Bergamo, Italy: Association for Computing Machinery, 2015, pp. 284–294. ISBN: 9781450336758. DOI: 10.1145/2786805.2786845. URL: https://doi.org/10.1145/2786805.2786845.