



**KIDELTA**  
**LEARNING**

Scalable AI for Automated Driving

Final Event | March 09, 2023

# Automotive Suitability

TP4 overview





1

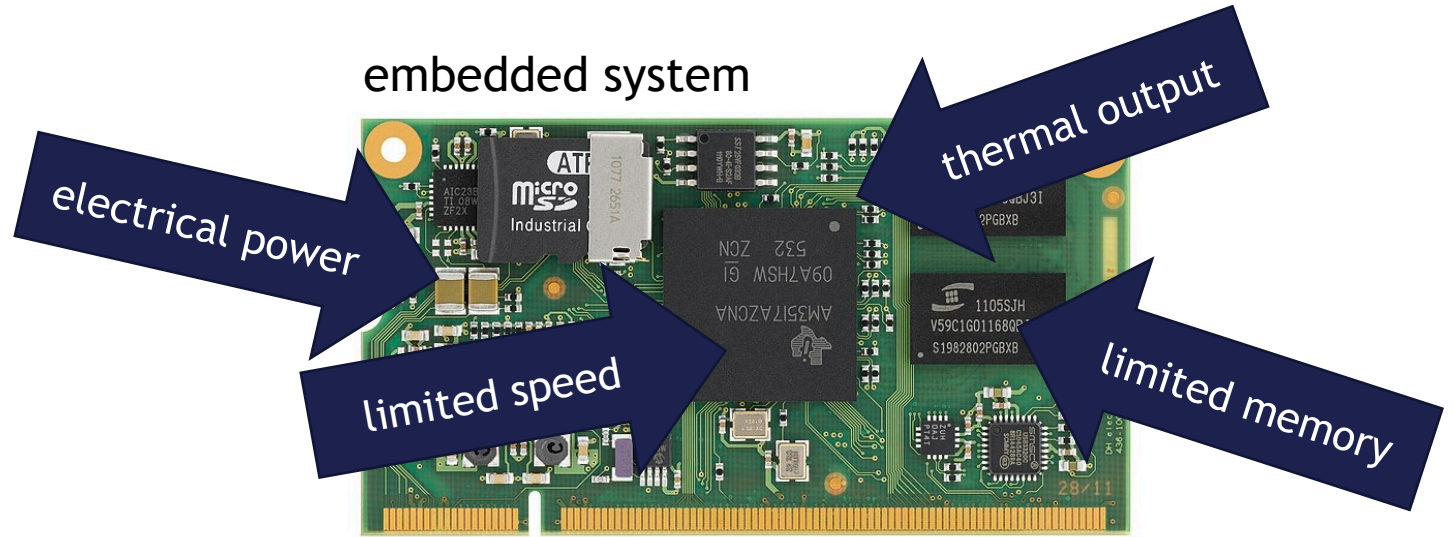
**Automotive suitability**

# Transferring AI from the Lab to the Car



Wikimedia commons, CC BY-SA 3.0

embedded system



Wikimedia commons, CC BY-SA 3.0



Jernej Furman, CC BY 2.0

# Transferring AI from the Lab to the Car

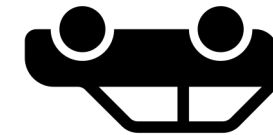
lab data (real/synthetic)



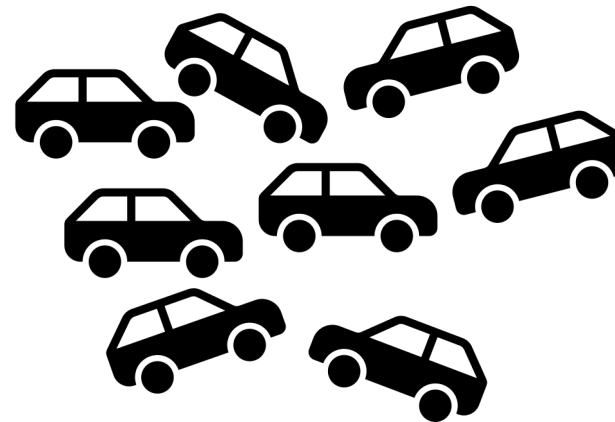
novel objects



abnormal behaviour

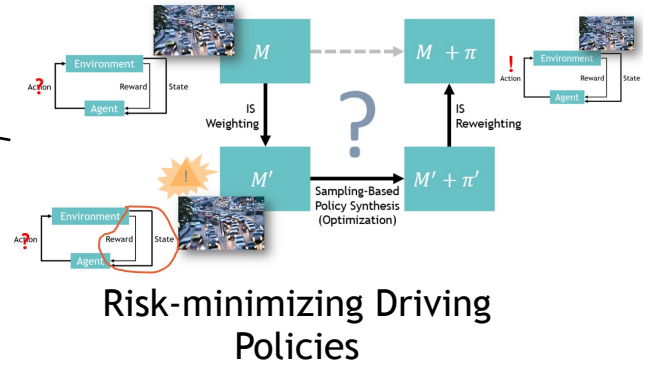
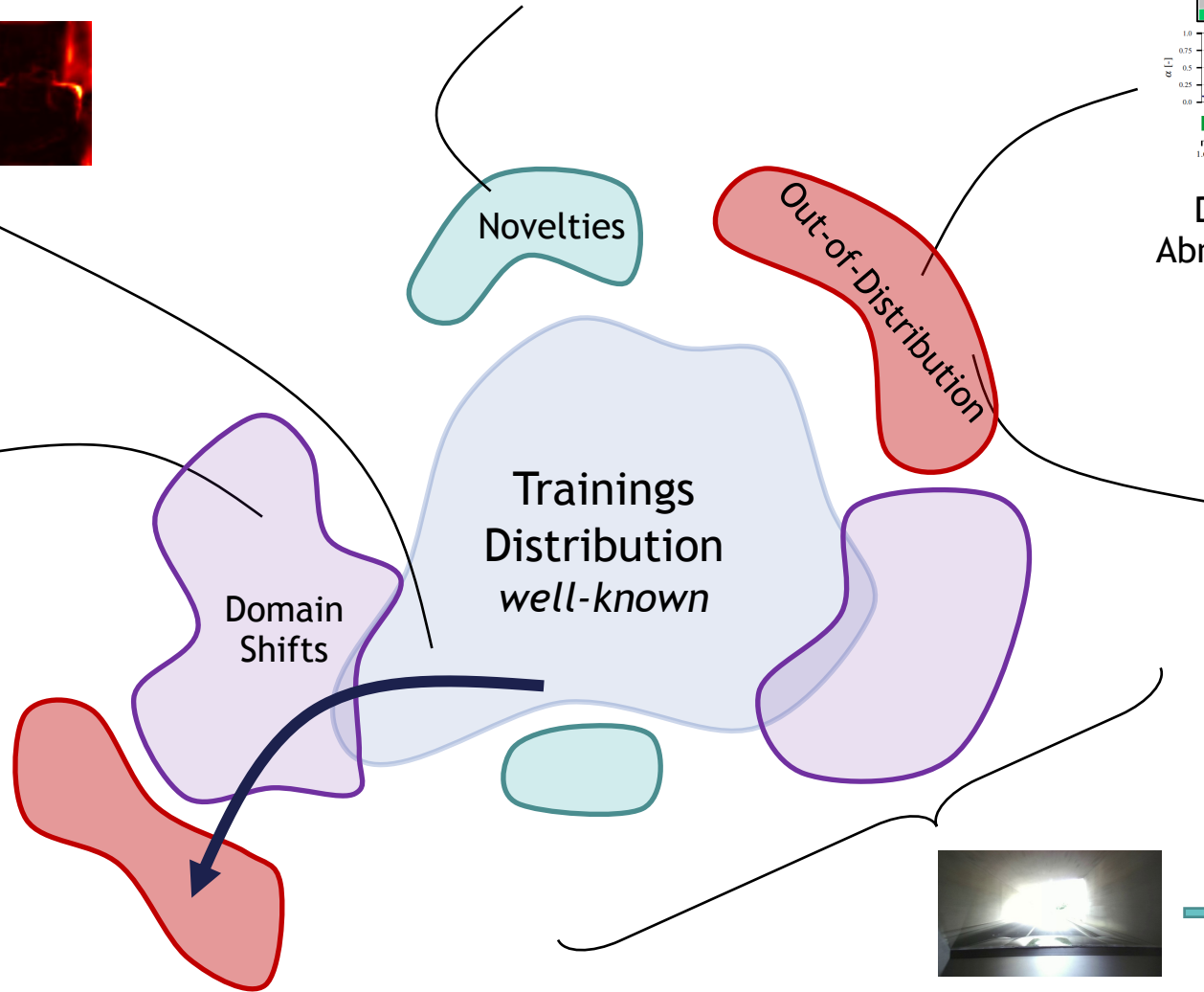
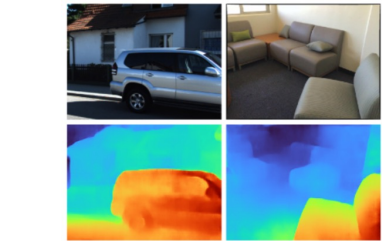
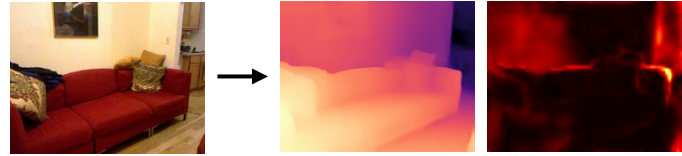
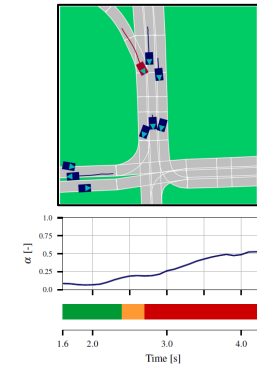
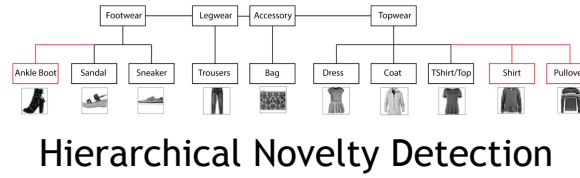


unexpected szenarios



Wikimedia commons, CC BY-SA 3.0

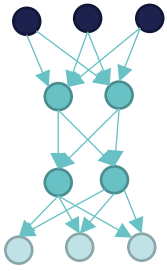
# The Real World Delta





# The Embedded Systems Delta

trained neural network



compiler

source code

```
# while ((next_nib = a_con_bin(*str)) < base)
atoui_loop:
    lb $1, 0($4)
    sw $1, 0($sp)
    jal a_con_bin
    addu $7, $1, $0
    sltu $1, $7, $0
    beqz $1, atoui_done

# if (accum > max_accum)
sgtu $1, $8, $9
bnez $1, atoui_overflow

# tmp = (accum * base) +
multu $1, $8, $6
addu $1, $1, $7

# if (tmp < accum)
sltu $7, $1, $8
bnez $7, atoui_overflow

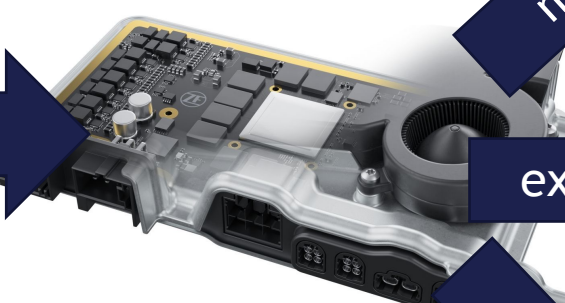
# accum = tmp
addu $8, $1, $0

# str++
addui $4, $4, 1

} atoui_loop
atoui_overflow:
.data
overflow_msg:
.asciz "\n"
.text
la $1, overflow_msg
sw $1, 0($sp)
jal pr
```

runs on

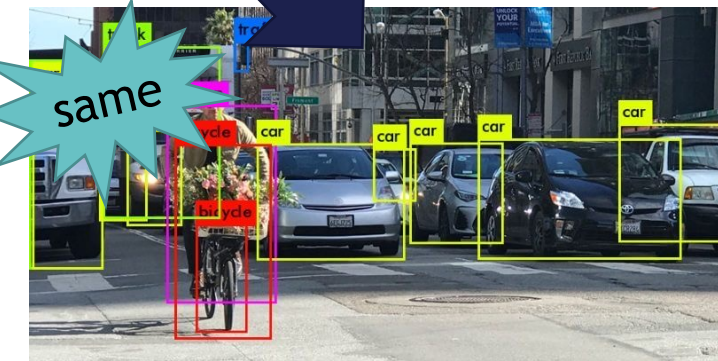
embedded system



exec. time



function



less

memory

less

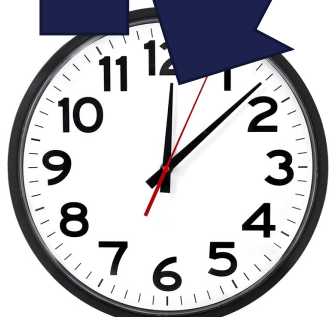
same

modify

predict

transform

compiler



# Highlight-Presentations



- Domenik Helms - AI Hardware Modeling
- Saqib Bukhari - Interpretable Pruning
- Philipp Schröppel - Robust Depth Estimation

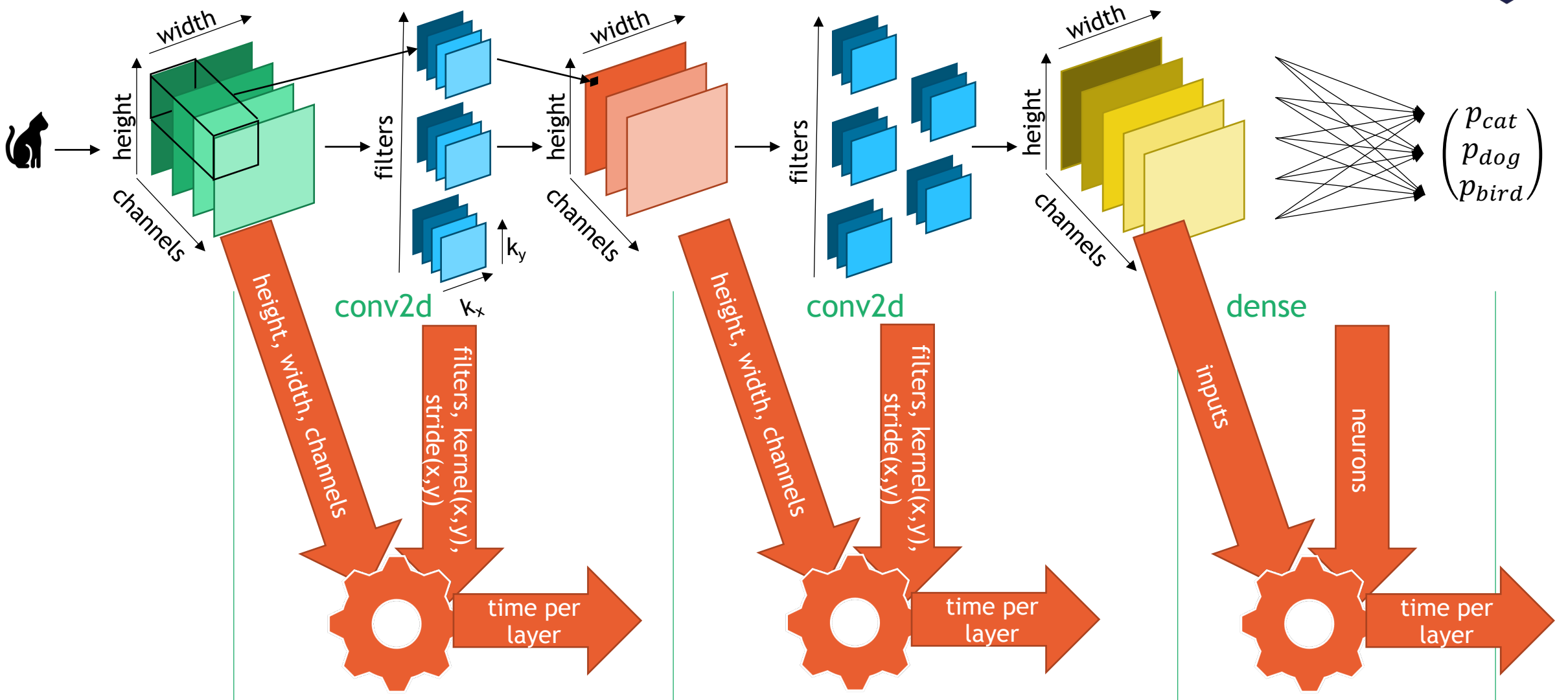
# 2



## Prediction



# Modeling idea



# Prediction of AI Properties



```
C:\Users\dhelms\AIM6\main.py - Notepad++
Datei Bearbeiten Suchen Ansicht Kodierung Sprachen Einstellungen Werkzeuge Makro Ausführen Erweiterungen F
main.py x aimodel.py x ailayer.py x modelCharacterizer_MyriadX.py x hw_timing_model.py x conv2D_model.py
290
291
292 # get AI benchmark from the repository
293 ai = tf.keras.applications.DenseNet121()
294
295 # initialize the aimodel withj the benchmark i
296 graph = aimodel.Aimodel(ai)
297
298 # print and write out report
299 graph.write_report ()
```

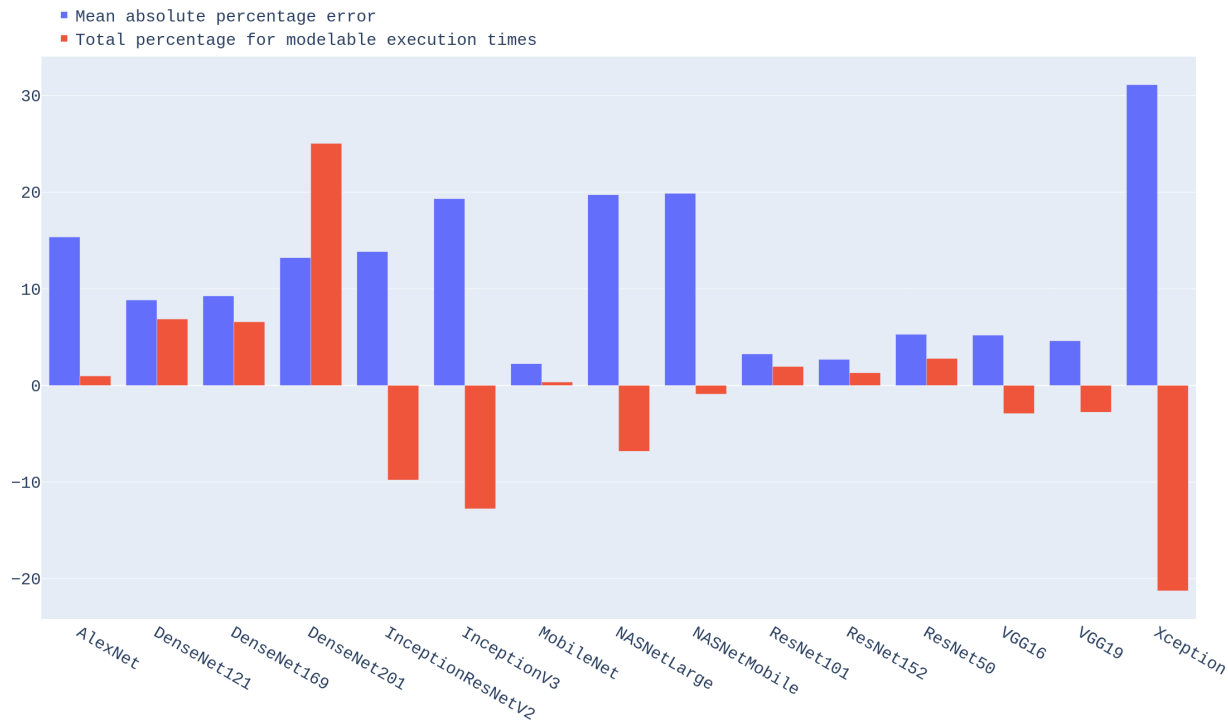


```
AIM6 - python main.py
Modelled time:
t=-2ms
General AI Data:
Inputs:
conv5_block16_1_bn/cond/Mem:0 (None, 7, 7, 128) conv5_block16_1_bn->
Outputs:
conv5_block16_1_relu (None, 7, 7, 128) conv5_block16_2_conv<-
Layer parameters:
activation: rel
Name: conv5_block16_2_ Type: Conv2D
Modelled time:
t=25.34117647058823ms
General AI Data:
Inputs:
conv5_block16_1_relu/Relu:0 (None, 7, 7, 128) conv5_block16_1_relu->
Outputs:
conv5_block16_2_conv/Conv2D:0 (None, 7, 7, 32) conv5_block16_concat<-
Data:
conv5_block16_2_conv/kernel:0 (3, 3, 128, 32) 36864 trainable parameters
Layer parameters:
filters: 32
kernel_size: (3, 3)
strides: (1, 1)
padding: same
activation: linear
Name: conv5_block16_concat Type: Concatenate
Full HW measurement:
t=0±0ms 0 HW layers
layer type:
exec type:
layer name:
Mode 0 per layer HW measurement:
t=0±0ms 0 HW layers
layer type:
```

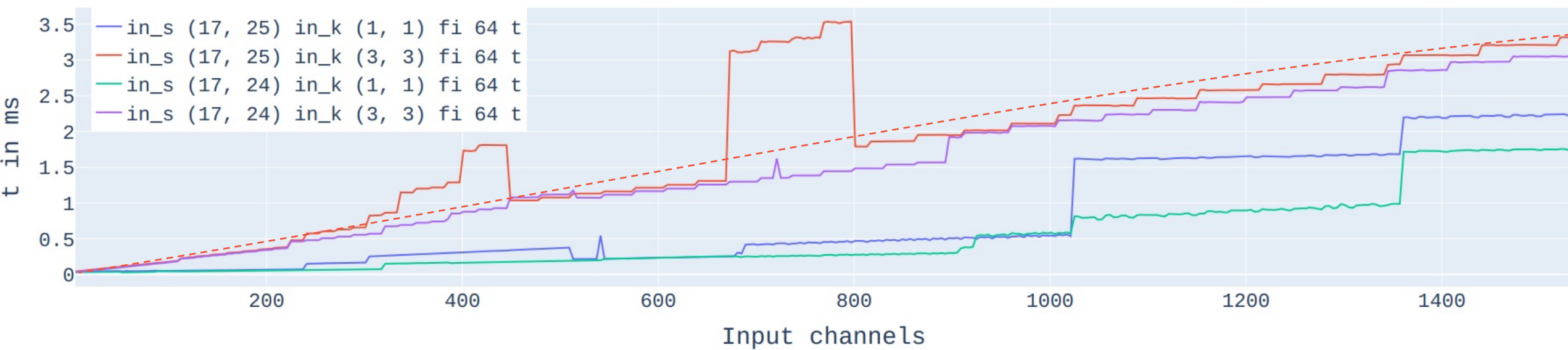
25ms

175kB

# Benchmark results neural compute stick



`in_s ([17, 17], [24, 25]), in_c [5, 1533], k [(1, 1), (3, 3)], fi [64, 64], st (1, 1)`

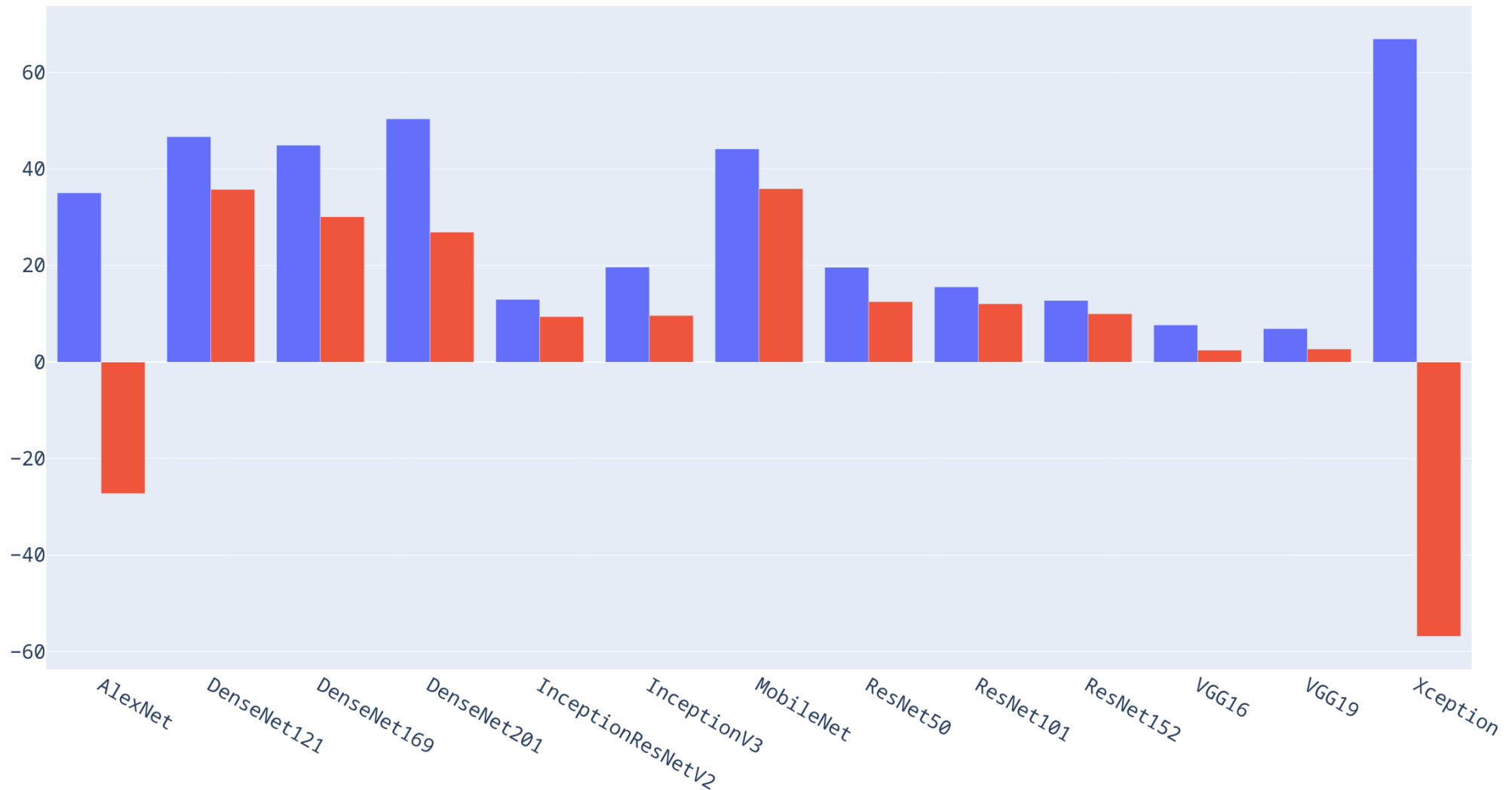






# Benchmark results GPU

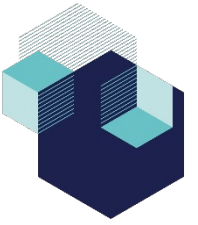
- Mean absolute percentage error
- Total percentage for modelable execution times



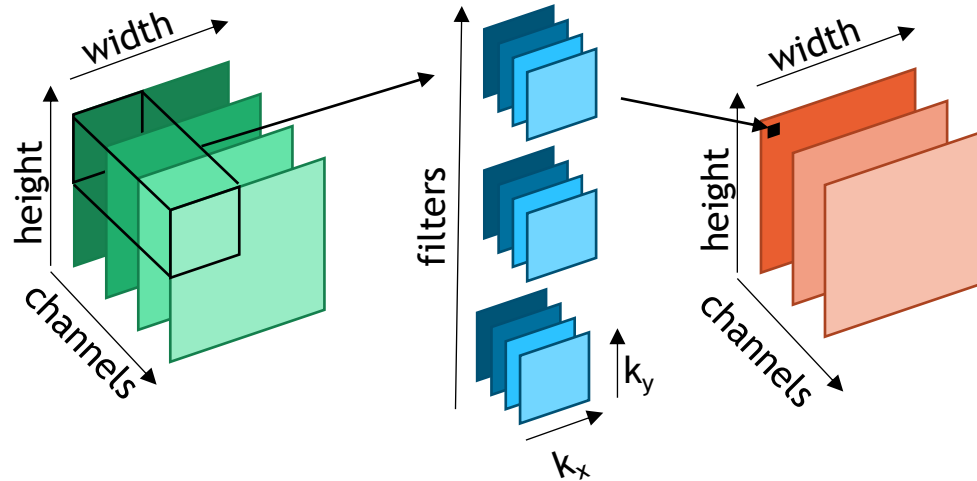
3



Optimization



# Tucker decomposition



$$y_f = \sum_{k_x, k_y, c} a_{k_x, k_y, c} \cdot W_{k_x, k_y, c, f} = A \times_4 W$$

$$W \in \mathbb{R}^{k_x \times k_y \times c \times f}$$

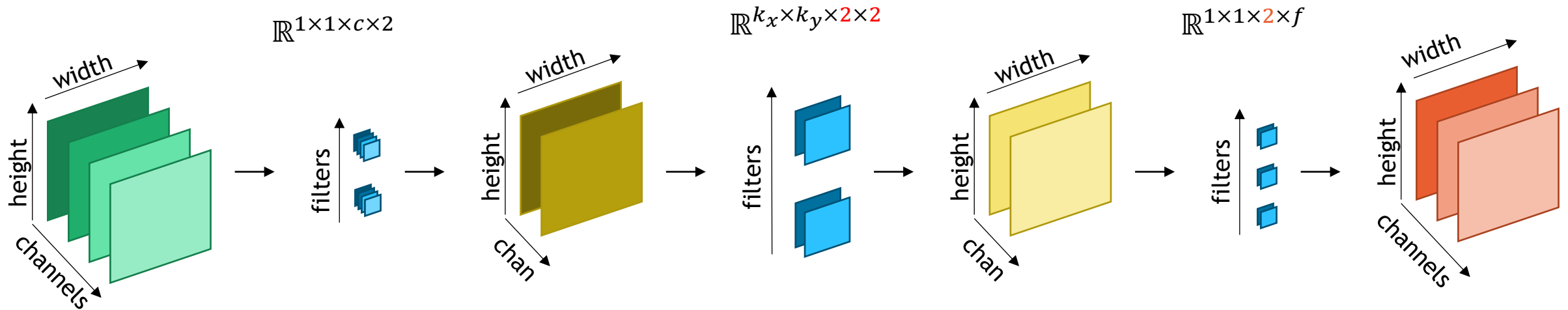
$$\text{Tucker}(p, q) \rightarrow W \approx C \times_3 G \times_4 F$$

$$G \in \mathbb{R}^{k_x \times k_y \times p \times q}$$

$$C \in \mathbb{R}^{1 \times 1 \times c \times p}$$

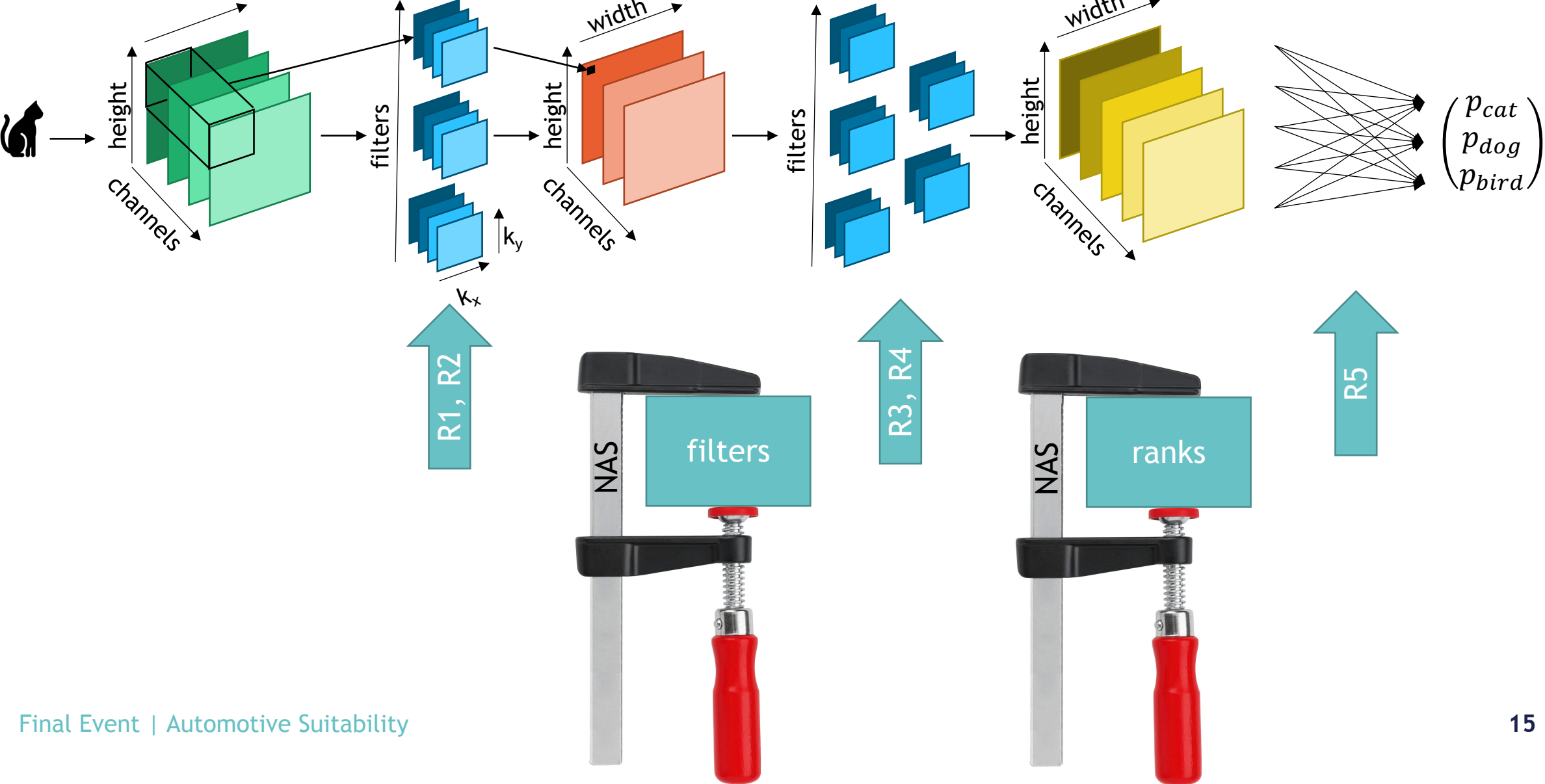
$$F \in \mathbb{R}^{1 \times 1 \times q \times f}$$

## Tucker(2,2)



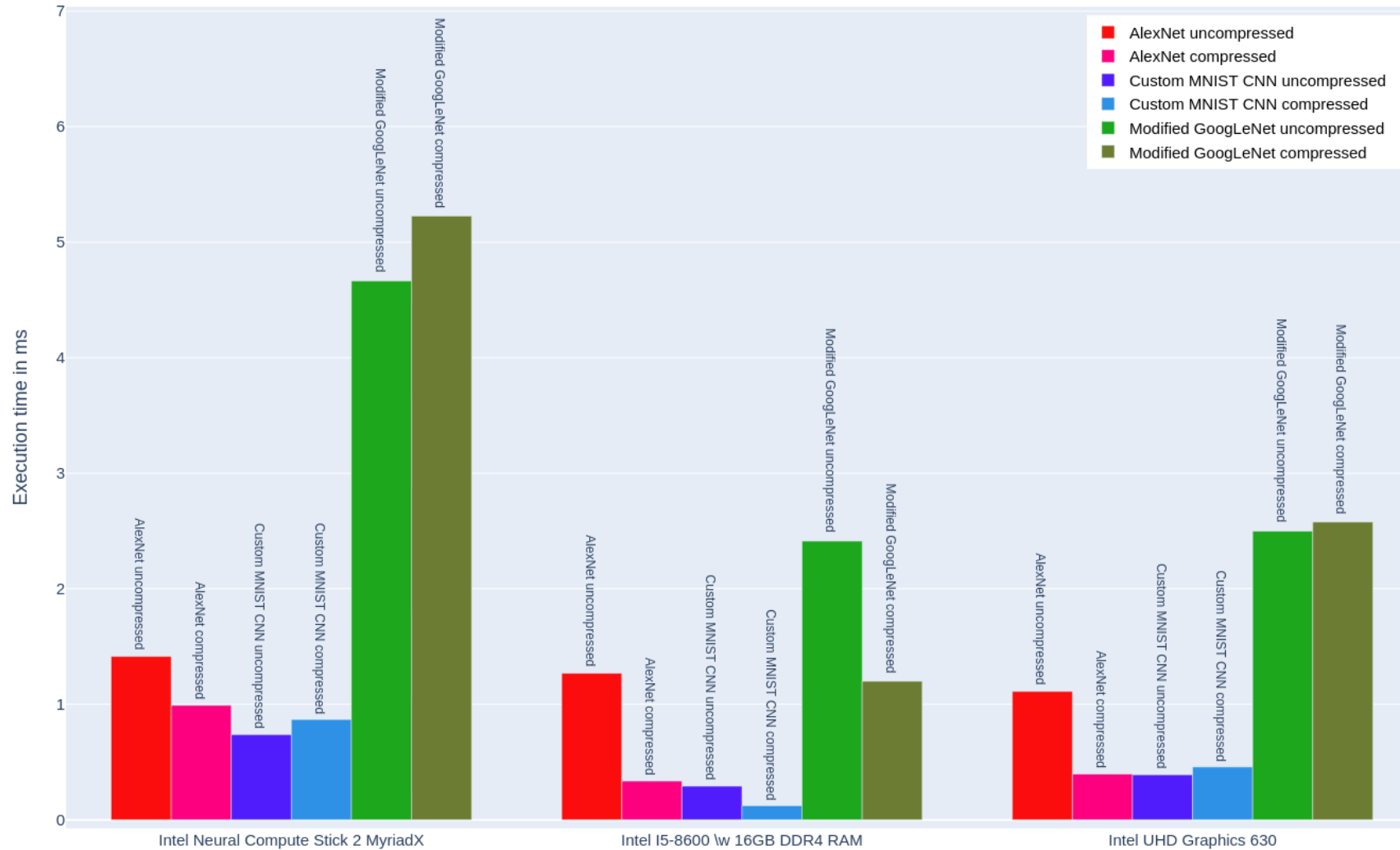


# Introducing NAS





# Execution time on various HW platforms







**KIDELTA**  
**LEARNING**

Scalable AI for Automated Driving

# Thank you for your attention

KI Delta Learning is a project of the KI Familie. It was initiated and developed by the VDA Leitinitiative autonomous and connected driving and is funded by the Federal Ministry for Economic Affairs and Climate Action.



**KI**  
**FAMILIE**

[www.ki-deltalearning.de](http://www.ki-deltalearning.de)  @KI\_Familie  KI Familie

Supported by:



on the basis of a decision  
by the German Bundestag