March 28-30, 2022 | San Francisco Bay Area
Ecosystem of tools for better productivity

Danilo PAU, Technical Director, IEEE and ST Fellow, STMicroelectronics
Global Developer Population and Demographic Study 2019, Vol 1

Source: https://www.daxx.com/blog/development-trends/number-software-developers-world
*https://www.stateofai2019.com/chapter-6-the-war-for-talent/#:~:text=Estimates%20of%20the%20number%20of,Al%20originated%20in%20academia.

- Python Developers: 7M (2018)
- Java Developers: 8M (Q3 2020)
- AI Developers: 22,000 to 350,000
- C/C++/C# Developers: 11.6M (2018)

Resource's GAP up to 3 order of magnitude
How to bridge the AI and embedded communities?
More FAEs? How many?  WRONG!

24.1 billion
IoT connected devices in 2030 ($7.6tn 2019)

$1.5 trillion
IoT revenue in 2030 ($456bn 2019)

Image source: Transforma Insights
Listen to their needs
Unstructured, sparse or partially labelled data

ML/NN Search & Optimization (AutoML, NAS, HPO)

Tiny ML/NN Optimizations for deployability

Interoperability with Deep Learning Frameworks (e.g. Tensorflow/Keras, Pytorch)

End-to-end Tools Ecosystem

AI back end tools (Code Gen, compilers, interpreters)

C/C++/C# Developers 11.6M (2018)

API back end tools

Developers 22,000 to 350,000

HL FRAMEWORKS

DL FRAMEWORKS

https://thispersondoesnotexist.com/
Interoperability

“The documented agreement reached by a group of individuals who recognize the advantage of all doing certain things in an agreed way”.

Leonardo Chiariglione

Scalability

Serving trillions of sensors

Automation

“everything that can be automated will be automated”.

First law

Shoshana ZUBOFF

Productivity

Keep calm and hand-craft ML

The needs
March 30, 2022 – 15:10 to 17:30

EON Tuner: AutoML for constrained devices
Jan JONGBOOM, CTO, Edge Impulse

Optimizing AutoML for the tinyML Future
Elias FALLON, VP for Machine Learning, Qeexo Co.

1 kB and not a bit more! The ideal weight for a tinyML model
Blair NEWMAN, CTO, Neutron

Model Optimization with QKeras’ Quantization-Aware Training and Vizier’s Automatic Neural Architecture Search
Daniele MORO, Software Engineer, Google

Automated Machine Learning under model’s deployability on tiny devices
Antonio CANDELIERI, Assistant Professor, University of Milano-Bicocca, Italy

Automating Model Optimization for Efficient Edge AI: from automated solutions to open-source toolkit
Dave Cheng, Qualcomm, USA

Session moderator: Danilo PAU
Technical Director, IEEE and ST Fellow, STMicroelectronics
Further challenges
• A sensor with a C compiler!
• QKeras importer
• Processor, 5 - 10 MHz
• Binary instructions
• Memory, 40 KiB
• μW energy envelope

• MCU standard vs custom ISA
• Code gen, interpreters
• Different compilers
• Processor, 10s - 100s MHz
• Embedded RAM, 10s-100s KiB
• Embedded FLASH, ≤ 2MB
• mW energy envelope
pip install qkeras==0.9.0

# feature extractor
x = x_in = Input(shape)
x = QActivation("quantized_bits(8, 7, alpha=1)", name="act_0") (x)
x = QConv2D(channel_CNN, (kernel_size_CNN, 1),
          kernel_quantizer="quantized_bits(8, 7, alpha=1)",
          use_bias = False,
          name="conv2d_1") (x)
x = BatchNormalization()(x)
x = QActivation("binary(alpha=1)", name="act_1") (x)
x = QConv2D(channel_CNN, (kernel_size_CNN, 1),
          kernel_quantizer="binary(alpha=1)",
          padding="same",
          use_bias = False,
          name="conv2d_2") (x)
x = MaxPooling2D(pool_size=(pool_size_CNN,1))(x)
x = BatchNormalization()(x)
x = QActivation("binary(alpha=1)") (x)
x = QConv2D(32, (1, 1),
          kernel_quantizer="binary(alpha=1)",
          padding="same",
          use_bias = False)(x)

# CNN_Head - classifier
x = Flatten()(x)
x = QDense(64, kernel_quantizer="binary(alpha=1)", use_bias=False)(x)
x = Dense(9, activation="softmax") (x)

Number of operation types in model:
sfmult_1_32: 305152 (8.5%)
smult_8_8: 95360 (2.6%)
sxor_1_1: 3204096 (89%)

Weight profiling:
conv2d_1_weights : 160 (8-bit unit)
conv2d_2_weights : 5120 (1-bit unit)
q_conv2d_weights : 1024 (1-bit unit)
q_dense_weights : 305152 (1-bit unit)
## Anomaly (bearings) classification

<table>
<thead>
<tr>
<th>Model</th>
<th>Accuracy (average of 10 trials) %</th>
<th>MACC</th>
<th>WEIGHTS KiB</th>
<th>RAM KiB</th>
<th>STM32 inference/s</th>
</tr>
</thead>
<tbody>
<tr>
<td>Keras</td>
<td>FP32</td>
<td>98.89</td>
<td>3,624,496</td>
<td>1218.56</td>
<td>75.5</td>
</tr>
<tr>
<td>TFLite</td>
<td>INT8</td>
<td>25.96</td>
<td>3,634,132</td>
<td>305.44</td>
<td>24.94</td>
</tr>
<tr>
<td>QKeras</td>
<td>INT1/FP32</td>
<td>98.39</td>
<td>3,624,432</td>
<td>41.32</td>
<td>18.88</td>
</tr>
</tbody>
</table>

Note: based on X-CUBE-AI (alpha version) code generation
Home Appliance classification

- Only current measurements
- 16 KHz sampling rate

!pip install qkeras==0.9.0

Number of operation types in model:
- smult_8_8: 276768 (3.4%)
- sxor_1_1: 7831008 (96.6%)

<table>
<thead>
<tr>
<th>Model</th>
<th>MACC</th>
<th>FLASH KiB</th>
<th>RAM KiB</th>
<th>Accuracy (%) K-fold=5</th>
<th>Inference/s</th>
</tr>
</thead>
<tbody>
<tr>
<td>Keras</td>
<td>FP32</td>
<td>5,864,688</td>
<td>108.5</td>
<td>99.43</td>
<td>12</td>
</tr>
<tr>
<td>TFLITE</td>
<td>INT8</td>
<td>5,888,972</td>
<td>27.68</td>
<td>99.39</td>
<td>30</td>
</tr>
<tr>
<td>QKeras</td>
<td>INT1/FP32</td>
<td>8,139,824</td>
<td>11.84</td>
<td>98.73</td>
<td>105</td>
</tr>
</tbody>
</table>

Note: based on X-CUBE-AI (alpha version) code generation
Average Accuracy of AutoEncoder model in 10 (CWRU) trials is: **98.5167**

**Number of operation types per inference:**

- `smux_1_16`: 7200
- `sxor_1_1`: 6080

**Weight profiling:**

- `q_conv2d_2_weights`: 40 (1-bit unit)
- `q_conv2d_3_weights`: 320 (1-bit unit)
- `q_conv2d_4_weights`: 320 (1-bit unit)
- `q_conv2d_5_weights`: 40 (1-bit unit)
- **Total**: 720 bits

**Activations profiling:**

- **Total allocation**: 512 bytes

Signed Qm.n

or

unsigned UQm.n
How to automate the dqnn design?

sensor \( p \)

\[ \text{signed } Q_{m\cdot n} \]

\[ \text{or} \]

\[ \text{unsigned } UQ_{m\cdot n} \]

\[ m \]

\[ n \]

\[ \text{deeply quantized neural network}_{qmn} \]
Our technology starts with You

danilo.pau@st.com
Copyright Notice

This presentation in this publication was presented as a tinyML® Summit 2022. The content reflects the opinion of the author(s) and their respective companies. The inclusion of presentations in this publication does not constitute an endorsement by tinyML Foundation or the sponsors.

There is no copyright protection claimed by this publication. However, each presentation is the work of the authors and their respective companies and may contain copyrighted material. As such, it is strongly encouraged that any use reflect proper acknowledgement to the appropriate source. Any questions regarding the use of any materials presented should be directed to the author(s) or their companies.

tinyML is a registered trademark of the tinyML Foundation.

www.tinyml.org