This work has received funding from the European Union's Horizon research and innovation programme under grant agreement No 101070374.

Towards a tailored mixed-precision sub-8-bit quantization scheme for Gated Recurrent Units using Genetic Algorithms

Riccardo Miccini[1,2], Alessandro Cerioli[1,2], Clément Laroche[1], Tobias Piechowiak[1], Jens Sparsø[2], and Luca Pezzarossa[2]

[1]GN Store Nord / [2]Technical University of Denmark

TinyML Research Symposium

April 22, 2024

Motivations and Contributions

- AI for audio processing on embedded devices
  - Sub-8-bit arithmetic can be quite efficient
  - Existing quantization tools have **limited support for sub-8-bit or GRUs**
  - Quantization parameters can be tedious to tune

- We propose:
  - Integer modular quantization scheme for GRUs
  - Genetic search for Pareto-optimal **any-bit quantization scheme**
  - Evaluation on sequence classification tasks of varying complexity

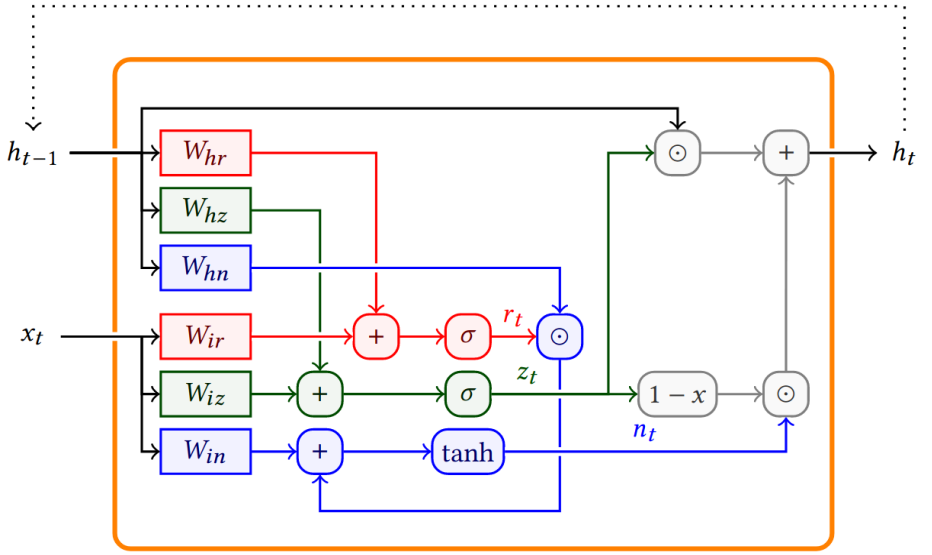
Outline

- Background
- Methods
- Results
- Conclusion

Background | Gated Recurrent Unit (GRU)

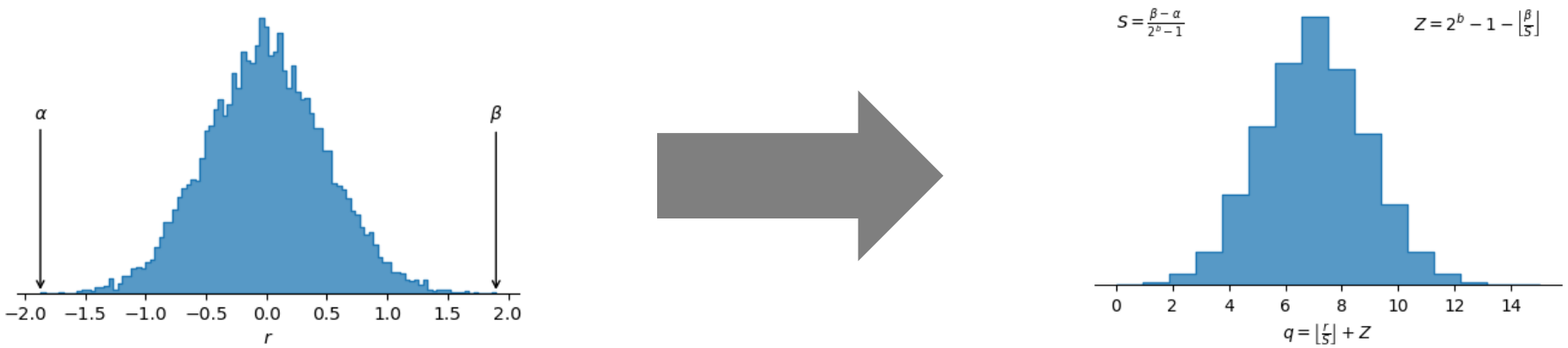
- GRUs are a class of Recurrent Neural Networks
  - Output depends on current input and previous output
  - Can **process sequential data** of arbitrary length
  - Often used in audio applications

- Like LSTMs, but more compact
  - Address vanishing/exploding gradient
  - 3 gates: reset, update, new state
  - Internal state is the output



Background | Quantization

- Map **continuous** data (high bit-width) **into** a **discrete** set (low bit-width)
  - Decrease storage, memory, and compute requirements
  - Lower precision might affect performances
- Linear/uniform quantization
  - Straightforward and most popular
  - Parameterized by bit-width, scaling factor S, and zero-point Z



Background | Genetic Algorithms (GA)

- Class of **optimization** algorithms
  - Inspired by biological principles (natural selection, evolution)
  - Suitable for non-differentiable multidimensional problems
- Start with a random population
  - Each specimen is a potential solution characterized by its **genome**
  - Solutions are **evaluated**, **ranked**, and **selected** for mating
  - New specimens are generated through **crossover** and **mutation**
  - Repeat with new specimens

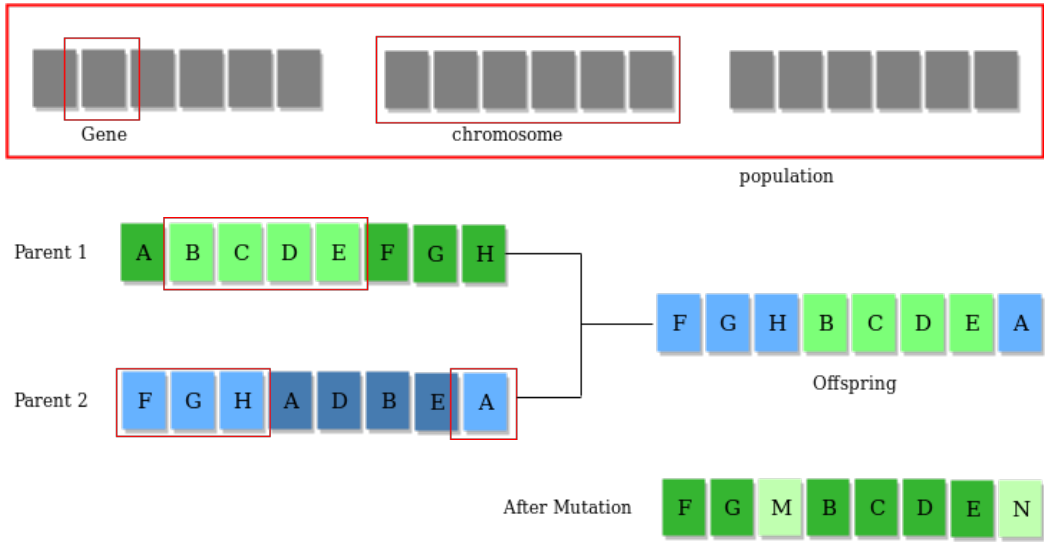
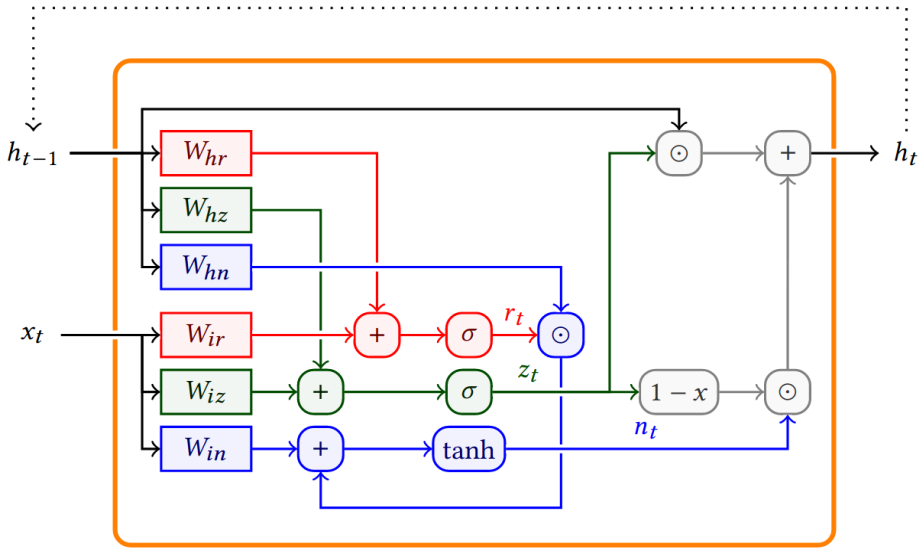


Image source: https://www.geeksforgeeks.org/genetic-algorithms

Methods | Modular quantization for GRUs

- Operators inside a GRU Cell
  - Linear/dense (matmul)
  - Hadamard (element-wise) product
  - Element-wise sum
  - Non-linearities: sigmoid, tanh
- Derive quantized versions
  - Substitute dequantization equation to the operator definition
  - Solve for quantized output
  - Combine scaling factors and convert to fixed-point
- For non-linearities, use look-up tables



$$\boldsymbol{y} = \boldsymbol{x_1} + \boldsymbol{x_2}$$

$$S_y(q_y - Z_y) = S_1(q_1 - Z_1) + S_2(q_2 - Z_2)$$

$$q_y = \frac{S_1}{S_y}(q_1 - Z_1 + \frac{S_2}{S_1}(q_2 - Z_2)) + Z_y$$

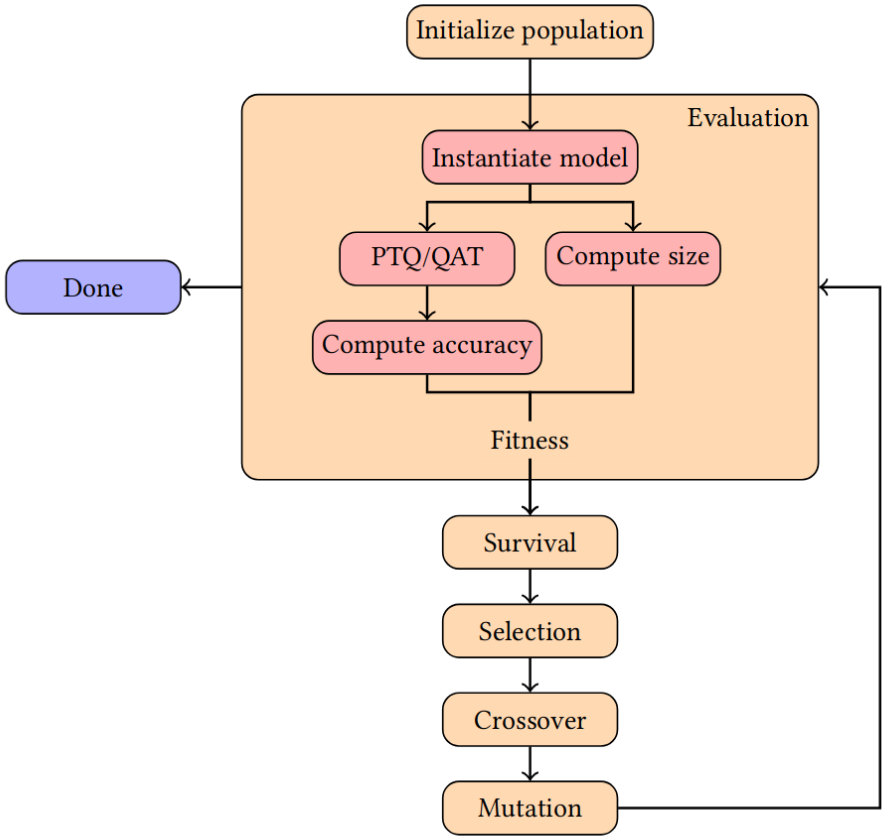
Methods | Quantization scheme search

- Genetic algorithm: NSGA-II
  - Selection step based on **Pareto efficiency**
  - Genome: bit-width (from 2 to 8 incl.) of each quantized operator, 17 in total
  - 40 initial specimens, 20 generations
- Fitness metrics
  - Model **accuracy**, computed on a validation set after quantization
  - Normalized complement of **model size**, based on number of trainable parameters

$$\widehat{\mathbb{M}}^{c} = 1 - \frac{\mathbb{M}_{Q}}{\mathbb{M}_{FP16}}$$

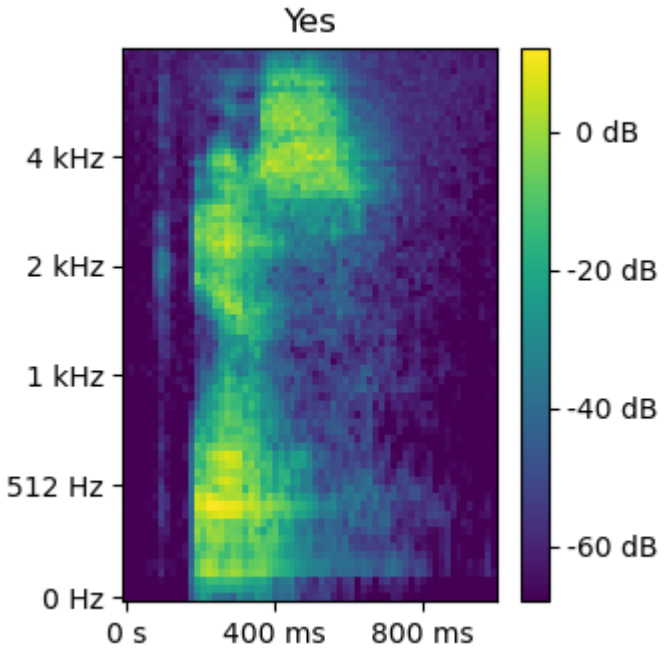
← model size with current quant. scheme

← model size with 16-bit weights

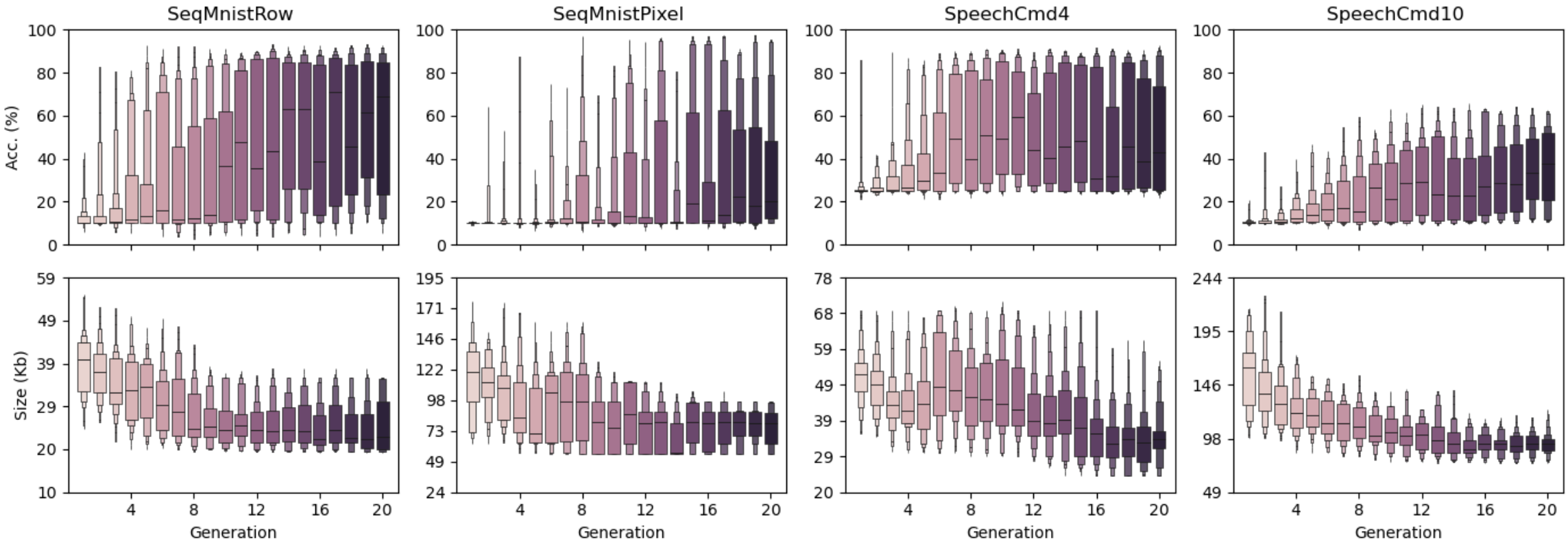


Results | Experimental setup

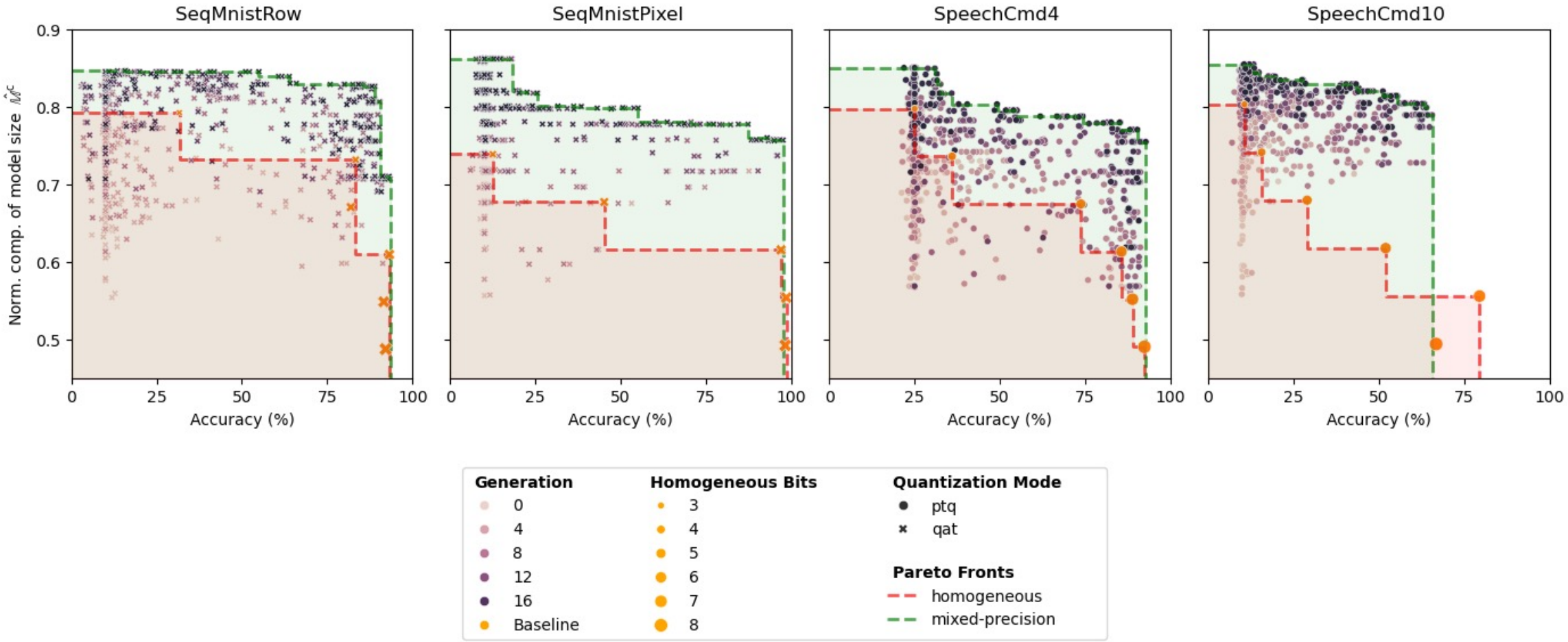
- Tasks and Datasets
  - MNIST (row-wise or pixel-wise): 28 timesteps × 28 features or 361 scalar timesteps
  - SpeechCmd (4 and 10 keywords): 63 timesteps × 40 features (Mel-Spectrogram)
- Model
  - GRU → FC → SoftMax; 256 or 128 hidden units
- Quantization
  - QAT on MNIST tasks (10% train set, learning rate = 1e-5)
  - PTQ on SpeechCmd tasks (100% train set for calibration)
- Baseline
  - Homogeneous quantization (bit-widths from 3 to 8 incl.)
- Evaluation metrics
  - Same as fitness metrics



Results | Genetic search



Results | Pareto fronts



Results | Observations

- Over the generations in the search:
  - Average accuracy increases
  - Model size decreases

- When using genetic search, we:
  - Exceed the Pareto fronts of baselines
  - Maintain 8-bit baseline performances
  - Achieve a model size reduction between 25% and 55%

- …except for SpeechCmd10
  - We experience a 17% drop in max accuracy
  - High bit-width solutions are not explored

Conclusion | Limitations and Future work

- Most of the derived solutions are low in accuracy
  - Constrain fitness metrics during survival step
  - Split search into exploration and exploitation phases

- Can it scale to more challenging scenarios?
  - Apply to larger tasks/models (e.g. speech enhancement)
  - Extend genome (e.g. observer types and granularity)

- Need for dedicated hardware support
  - Some SoCs might support this (e.g. GAP9 vectorized 2/4-bit arithmetic)

Conclusion | Summary

- In this work, we:
  - Introduce a modular integer quantization scheme for GRUs
  - Apply a multi-objective genetic search of quantization parameters
  - Evaluate the system on sequence classification tasks

- Results:
  - Heterogeneous solutions are more Pareto-efficient
  - Substantial decrease in model size with comparable accuracy

Copyright Notice

www.tinyml.org