Introduction to optimization algorithms to compress neural networks

Marcus Rüb

Marcus.rueb@hahn-schickard.de
https://www.linkedin.com/in/marcus-rüb-3b07071b2
Hahn-Schickard Villingen-Schwenningen

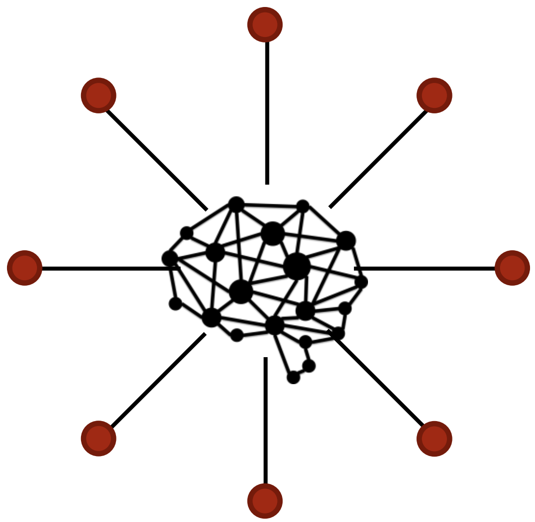
Agenda

Hahn Schickard

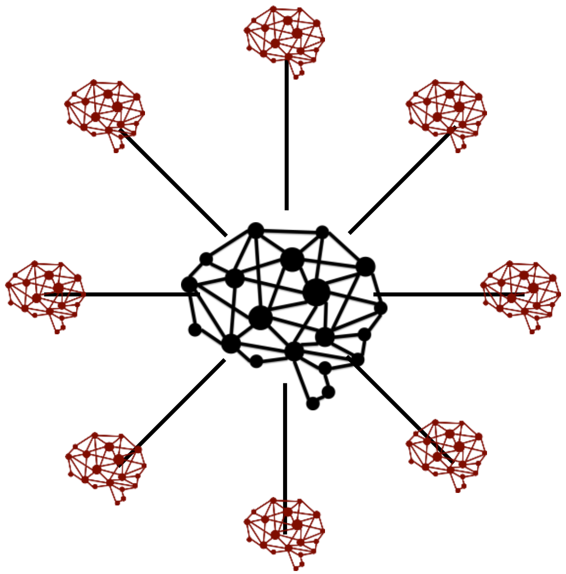
- What is tinyML and why do we need this?
- Quantization
- Knowledge distillation
- Pruning
- Other methods
- Take away

What is tinyML(Edge AI) and why do I need this?

Cloud AI



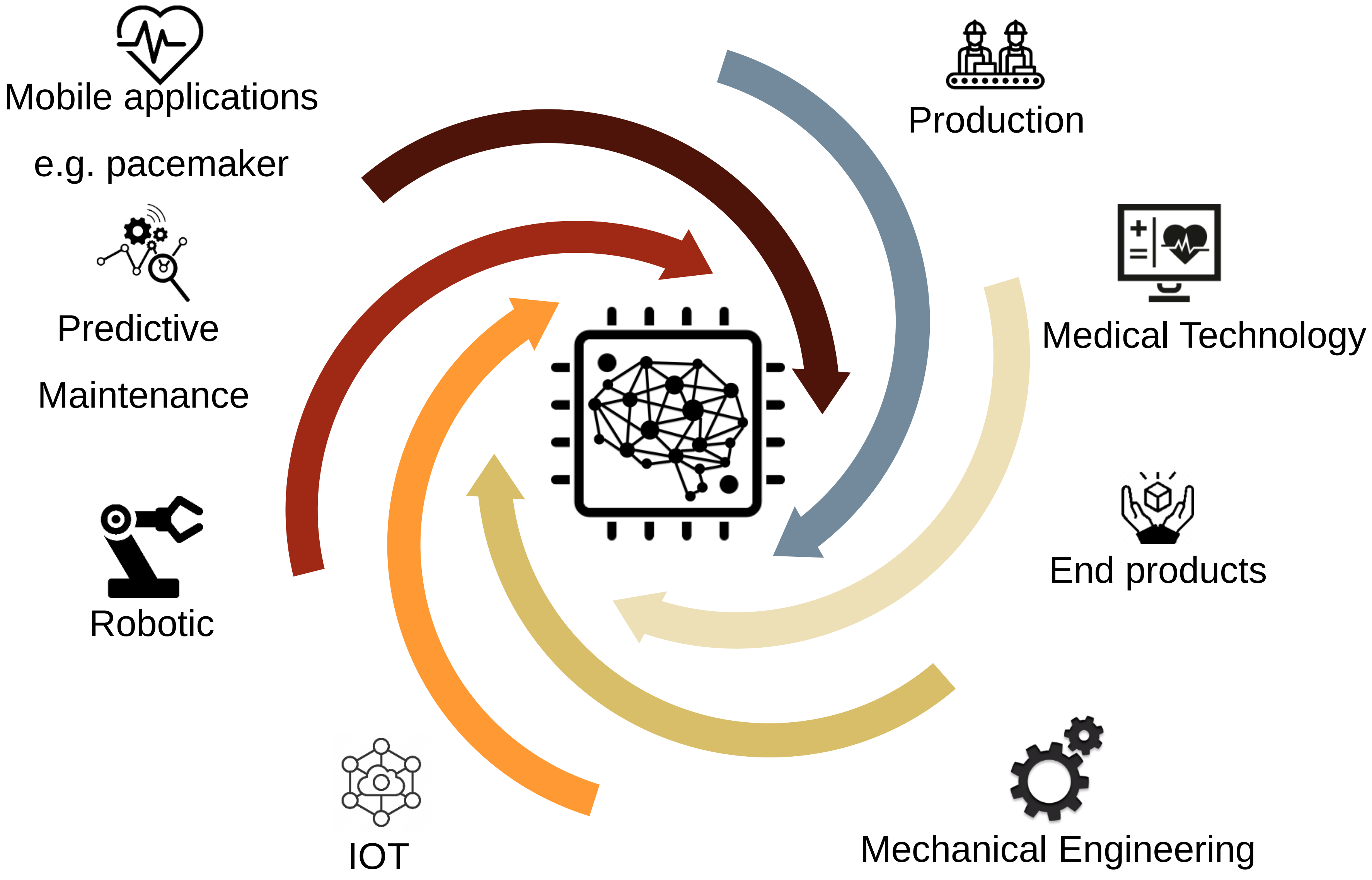
Edge AI



Benefits:

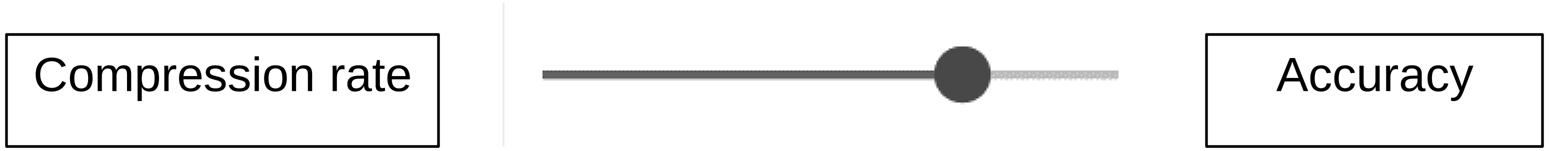
- **Privacy**
- **Low latency**
- **Energy saving**
- **Less communication**

Fields of application



Compress neural networks

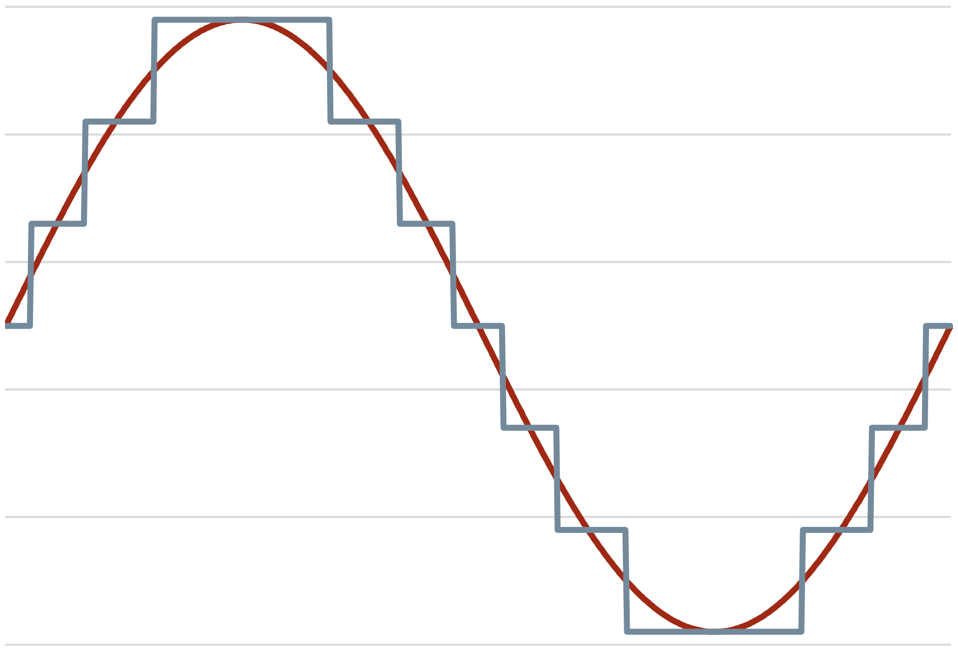
- The problems get complexer
- The models get bigger
- Solution: to compress the model
- Problem of compression: we get a trade-off



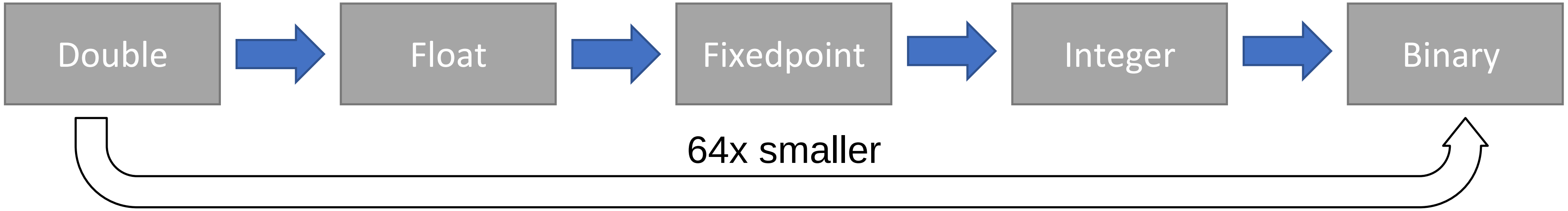
Quantization

- **Quantization** is the process of constraining an input from a continuous or otherwise large set of values (such as the real numbers) to a discrete set (such as the integers).*

*Wikipedia



Quantization



2,09	-0,98	1,02	0,09
0,05	-0,14	-1,08	2,12
-0,91	1,92	0	-1,03
1,87	0	1,03	0,98

2,09	2,12	1,92	1,87
0,05	-0,14	0	0,09
-0,91	-0,98	-1,08	-1,03
1,02	1,03	0,98	

2
0
-1
1

Retrain

Huffman coding

- Special case of quantization
- Make the model smaller but increase the inference time
- Can be good for Hardware implementations

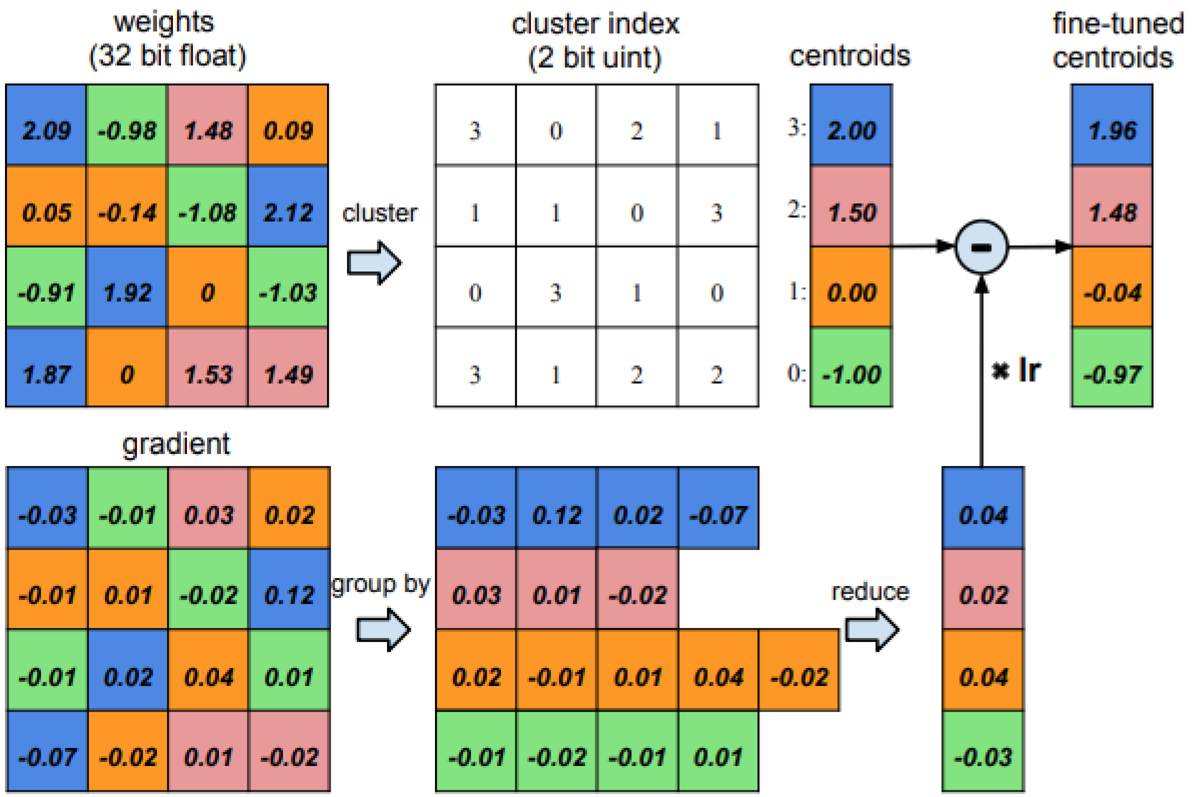


Figure 3: Weight sharing by scalar quantization (top) and centroids fine-tuning (bottom).

Quantization

- Pros:
  - Quantization can be applied both during and after training
  - Can be applied on all layer types
  - Can improve the inference time/ model size vs accuracy tradeoff for a given architecture
- Cons:
  - Quantized weights make neural networks harder to converge. A smaller learning rate is needed to ensure the network to have good performance.
  - Quantized weights make back-propagation infeasible since gradient cannot back-propagate through discrete neurons. Approximation methods are needed to estimate the gradients of the loss function with respect to the input of the discrete neurons.

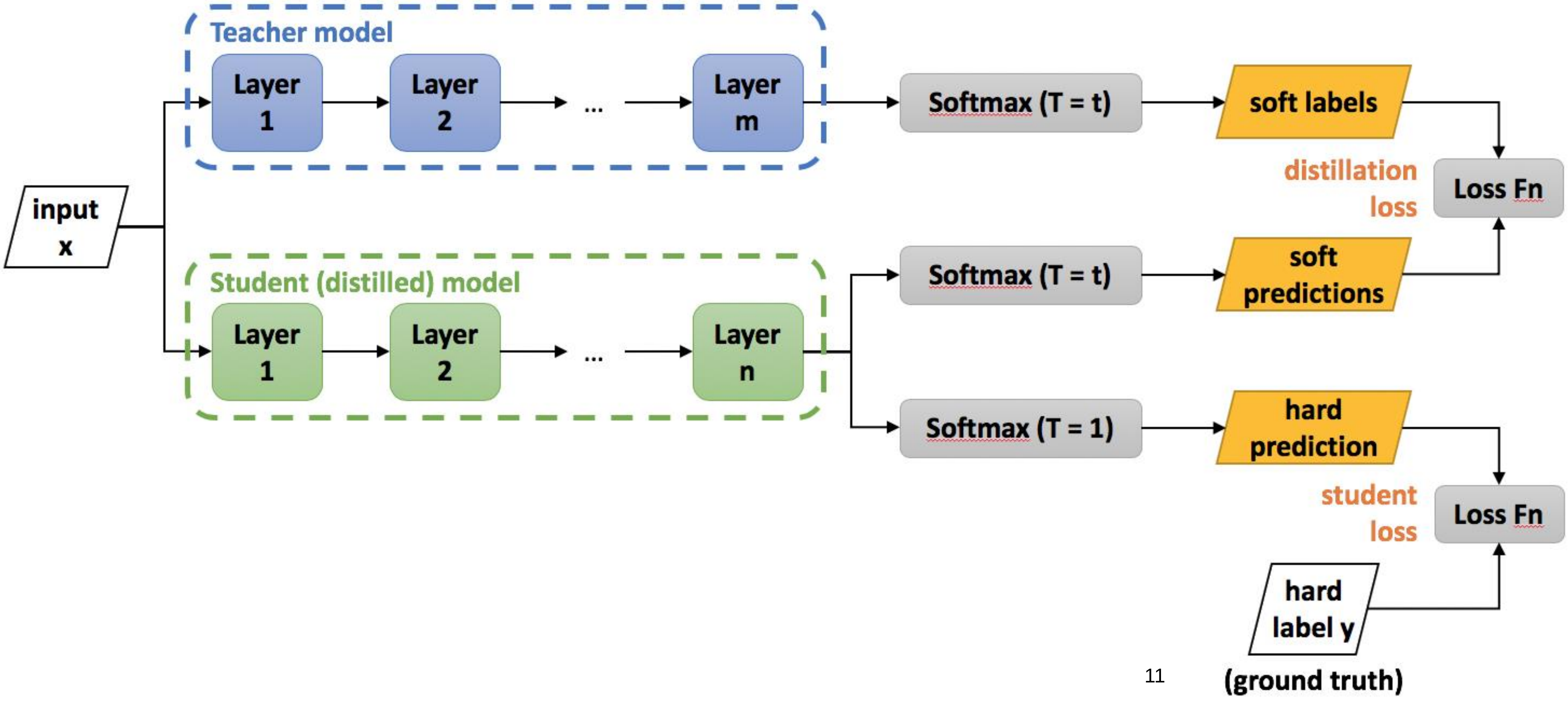
Dive deeper?

https://arxiv.org/pdf/1808.04752.pdf

https://www.tensorflow.org/lite/performance/post_training_integer_quant

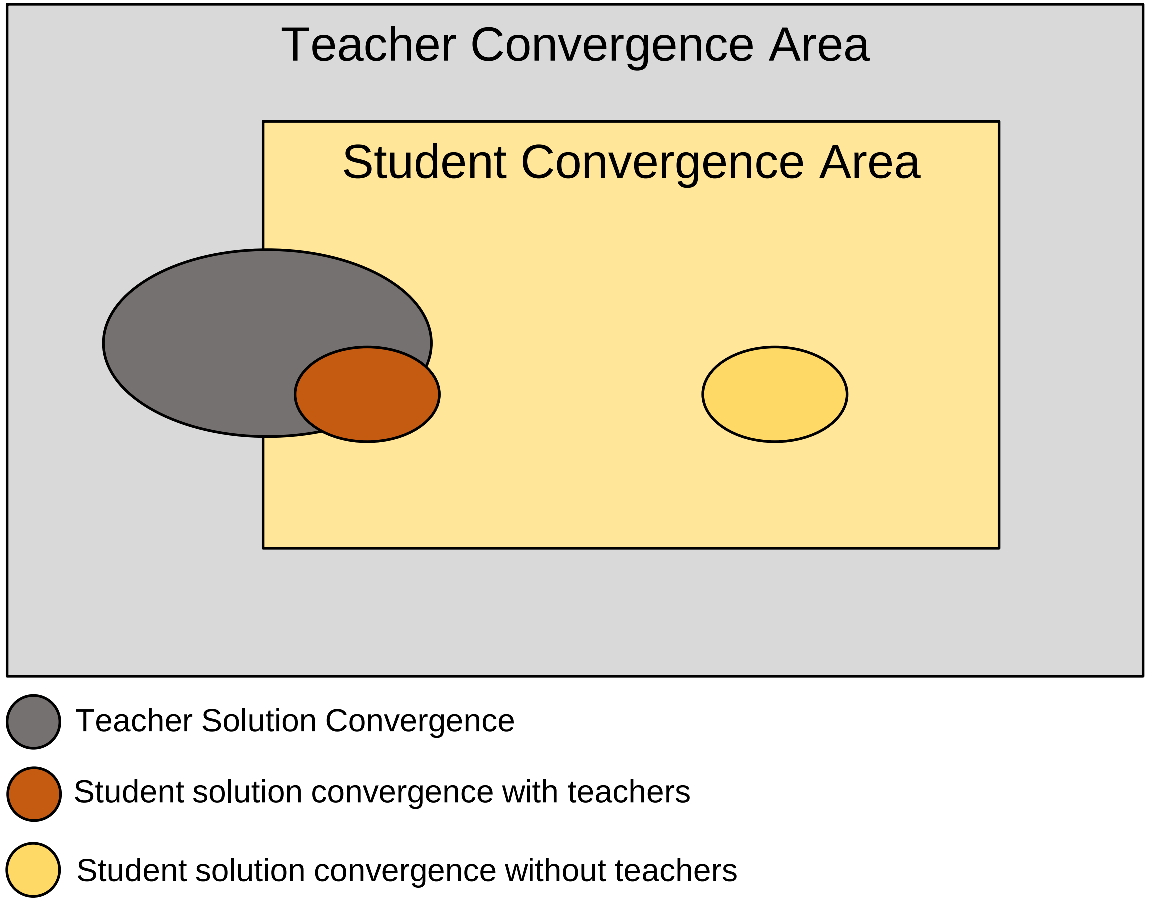
https://github.com/google/qkeras

Knowledge distillation



Knowledge distillation

- The teacher network guides the student network
- Up to 20x smaller networks



Knowledge distillation

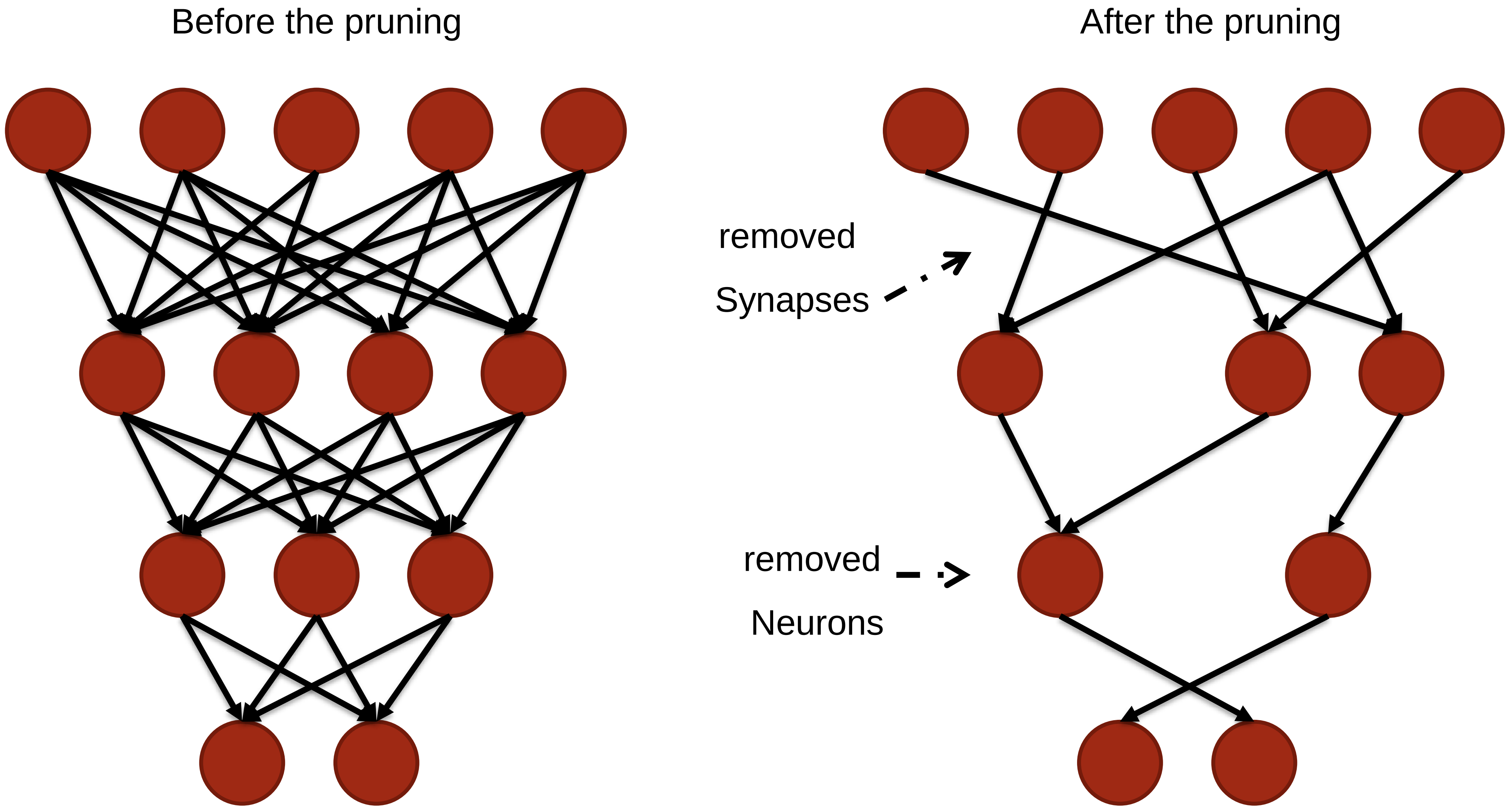
- Pros:
  - If you have a pre-trained teacher network, less training data required to train the smaller (student) network.
  - If you have a pre-trained teacher network, training of the smaller (student) network is faster.
  - Can downsize a network regardless of the structural difference between the teacher and the student network.
- Cons:
  - If you do not have a pre-trained teacher network, it may require a larger dataset and take more time to train it.
  - A good hyper-parameter set is hard to find.

Dive deeper?

https://arxiv.org/pdf/2006.05525.pdf

https://github.com/TropComplique/knowledge-distillation-keras

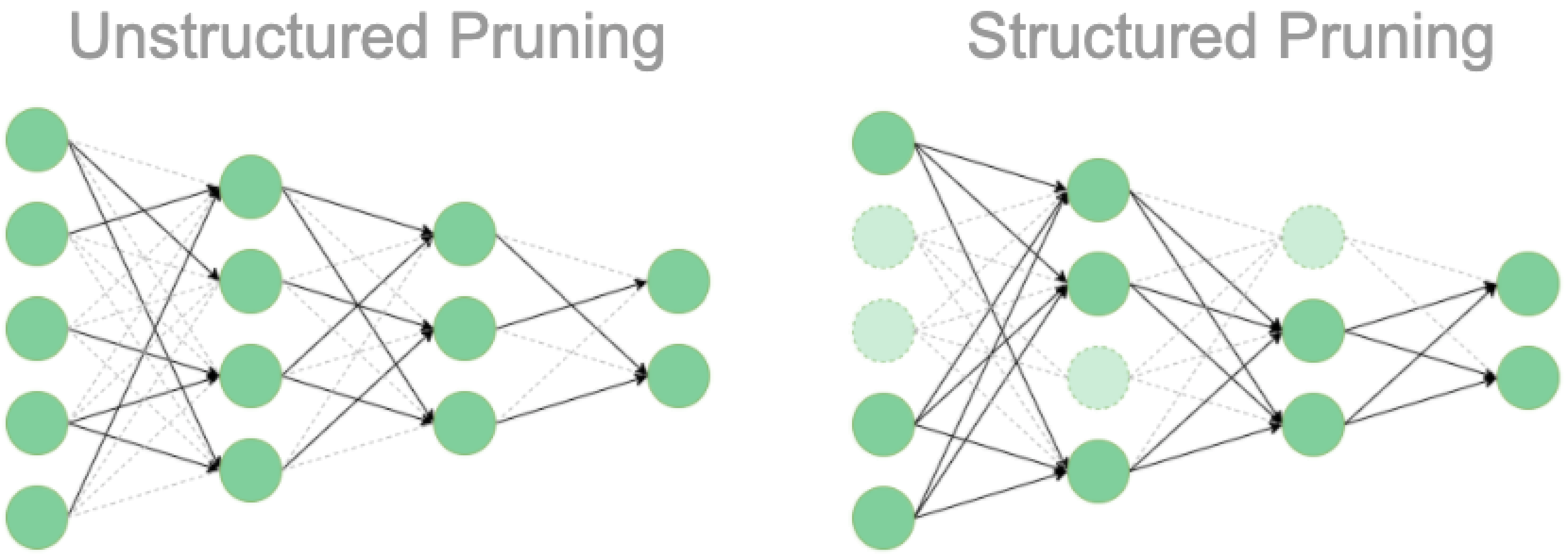
Pruning



Before the pruning
After the pruning
removed
Synapses
removed
Neurons

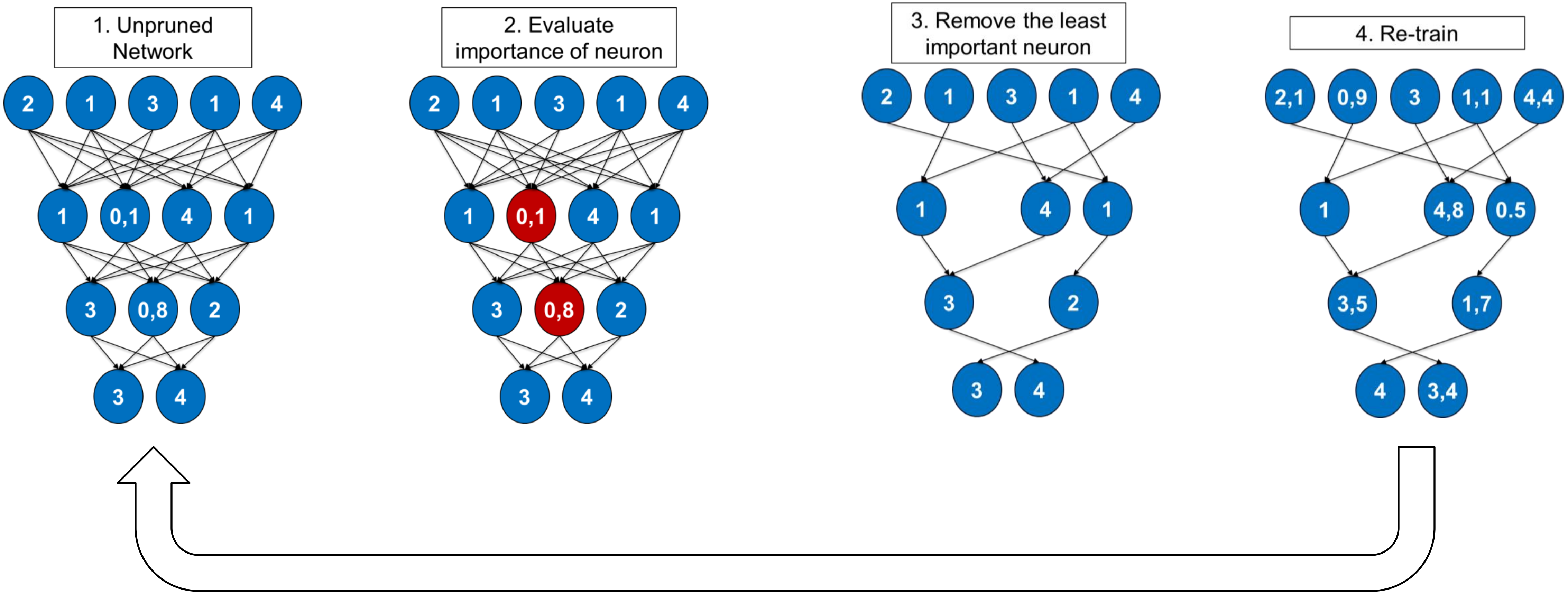
Structured pruning vs. Unstructured pruning

- Unstrutured pruning: delete connections between neurons
  - Benefit: easy to implement
- Strutured pruning: delete the whole neuron
  - Benefit: compress and speedup the model



Pruning process

2 to 13x smaller



How to know which Conections/neurons to prune?

- L1/L2 mean
- Magnitude
- Mean activations
- The number of times a neuron was zero on some validation set
- Matrix similarity

Pros and Cons

- Pros:
  - Can be applied during or after training
  - Can improve the inference time/ model size vs accuracy tradeoff for a given architecture
  - Can be applied to both convolutional and fully connected layers
  - Better generalization
  - Privacy preserving networks

- Cons:
  - Unstructured pruning does not speed up the inference

Dive deeper?

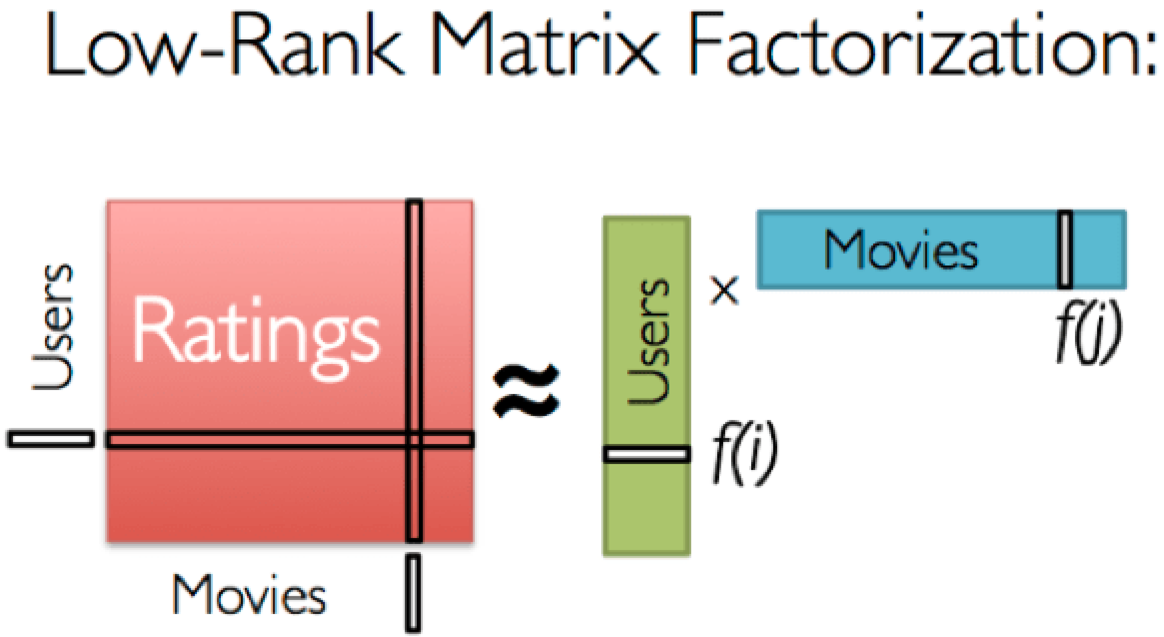
https://arxiv.org/pdf/1808.04752.pdf

https://www.tensorflow.org/lite/performance/post_training_integer_quant

https://github.com/Hahn-Schickard/Automatic-Structured-Pruning

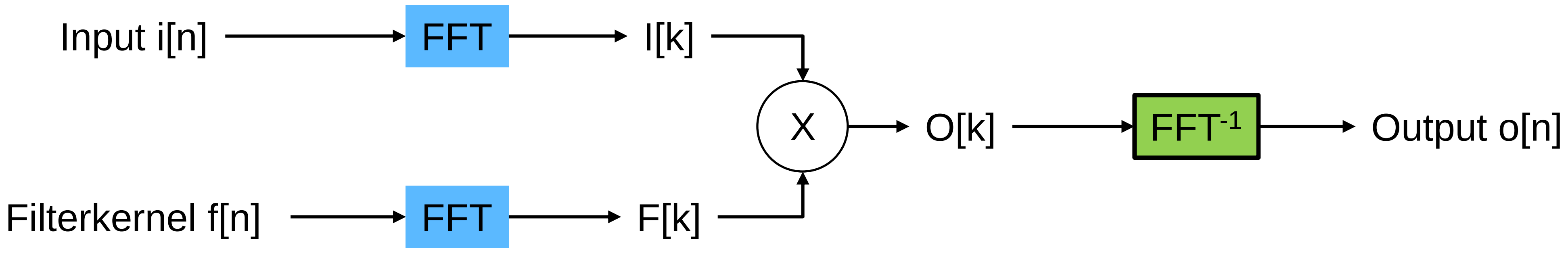
Low-rank factorization

- Done by an SVD
  - Singular Value Decomposition (SVD) of a matrix is a factorization of that matrix into three matrices
- The weight matrix get split into two vectors
- Con: Decomposition is a computationally expensive task



Fast-Conv

- Instead of calculate the convolution, calculate transform the input into the frequency-domain and calculate a multiplication
- The Filter kernel are pre-transformed
- Special case: Winograd-convolution -> faster, but only with even number of filterkernelsize
- Good for hardware implementations



Selective attention network

- „Divide et impera“ - divide and conquer
- Two algorithm:
  - The first select the area of interest
  - The other is the neural network

Summary

- We learned three compression methods
  - Quantization
    - Huffman coding
  - Knowledge distillation
  - Pruning
  - Low-rank factorization
  - Fast-Conv
  - Selective attention network

- Network compression work
- We can compress the model up to 20x of the size