KalEdge Lite

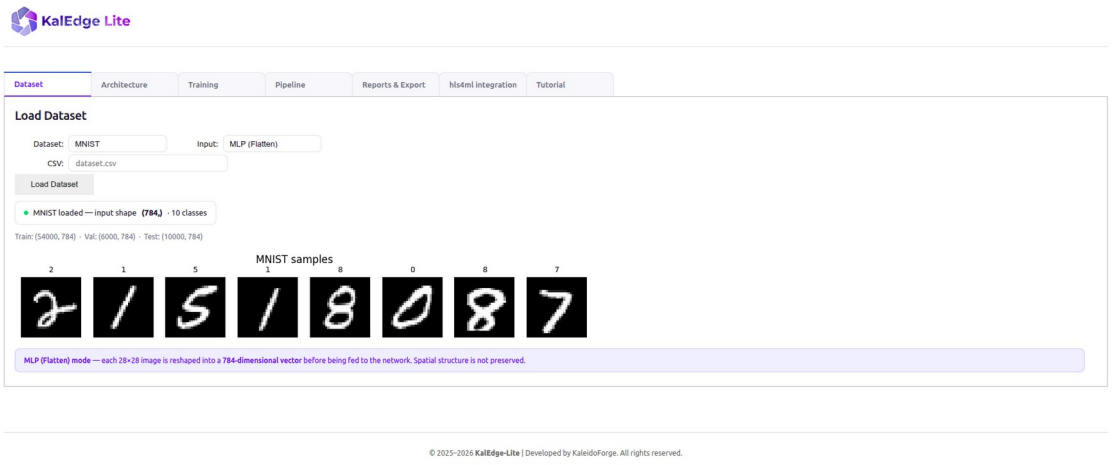
Training & Compression

Hands-on Exercises with KalEdge Lite

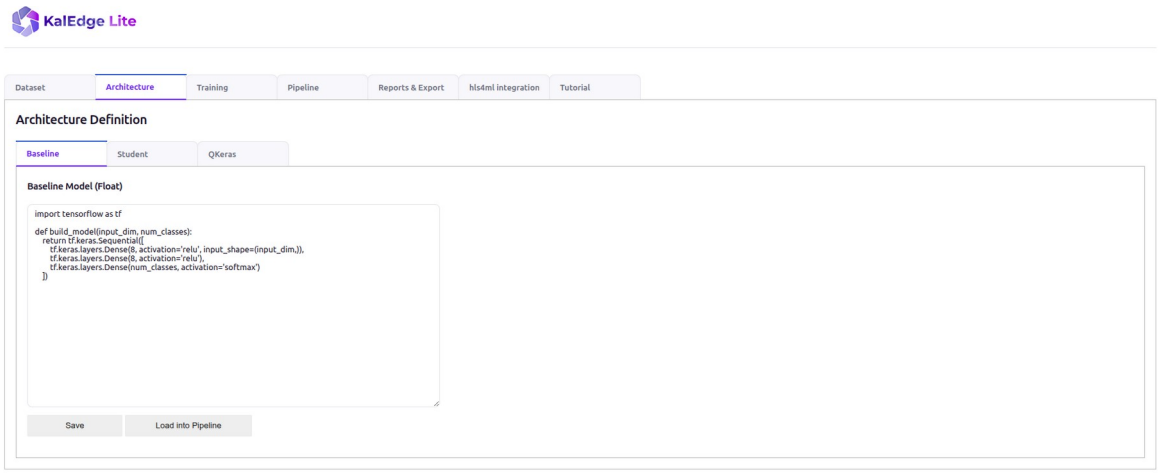
Use the standalone training tabs to complete each exercise.

Part 1: Baseline Training

We will use the MNIST dataset. Set the Input dropdown to MLP (Flatten) (this flattens each 28×28 image into a 784-dimensional vector before feeding it to the network).



Once loaded, go to the Architecture tab, select the **Baseline** model, click **Save** and then **Load into Pipeline** before starting training.



1.1. Effect of learning rate

Train the baseline model three times with different learning rates, keeping all other parameters fixed (epochs = 20, batch = 32, optimizer = Adam).

- Run 1: LR = 1e-2
- Run 2: LR = 1e-3
- Run 3: LR = 1e-4

Q1 Which learning rate converged fastest? Which one produced the best final validation accuracy?

Tip Watch the Loss curve, a zigzag pattern means LR is too high. A nearly flat curve at the start means LR is too low.

1.2. Epochs and overfitting

Train with LR = 1e-3, batch = 32, Adam, and experiment with epochs.

- Run 1: epochs = 10
- Run 2: epochs = 50
- Run 3: epochs = 100

Q2 At what epoch does the gap between train accuracy and val accuracy start to grow? What does this indicate?

Tip In KalEdge Lite the Accuracy plot shows both curves. Look for the point where the blue (val) line flattens while the purple (train) line continues rising.

1.3. Optimizer comparison

Train with LR = 1e-3, epochs = 30, batch = 32. Compare optimizers:

- Adam
- SGD
- RMSprop

Q3 Which optimizer reached the highest validation accuracy? Which was the most stable (least oscillation in the loss curve)?

Try it Try SGD with momentum = 0.9 (it often catches up to Adam if you increase epochs)

Part 2: Pruning

Use the **Training → Pruning tab**. Make sure you selected **From Scratch** mode in the **Pruning** tab.

2.1. Sparsity vs accuracy tradeoff

Train the baseline (30 epochs, Adam, LR = 1e-3). Then apply pruning with different sparsity levels:

- Sparsity = 0.3 (30% of weights zeroed)
- Sparsity = 0.5 (50%)
- Sparsity = 0.8 (80%)

Q4 At what sparsity level does the accuracy drop significantly? Is there a sparsity level where accuracy is preserved but the model is lighter?

Tip Check the Sparsity metric in the model metrics panel after training — it shows the actual percentage of zero weights in the final model.

2.2. Effect of training epochs with pruning schedule

Fix sparsity = 0.5 and vary the number of training epochs:

- Epochs = 5
- Epochs = 15
- Epochs = 30

Q5 How does the number of epochs affect the final accuracy when training with a pruning schedule? Is there a minimum number of epochs needed for the pruning schedule to reach the target sparsity?

Tip In KalEdge Lite, pruning and training happen simultaneously using a PolynomialDecay schedule. More epochs = more time for the model to adapt to the zeroed weights. If Sparsity in the metrics panel is lower than configured, increase epochs.

Part 3: QAT / QAP with Qkeras

Before starting:

1. Go to the **Architecture** tab → **QKeras** editor
2. Write or load your QKeras model, click **Save** then **Load into Pipeline**
3. Go to **Training** → **QAP (QKeras)** tab
4. Toggle the source button to **External model** (turns blue/violet)
5. Configure bits, int bits, epochs, LR and sparsity as indicated in each exercise

3.1. Bit-width and accuracy

Run **QAT (QKeras)** with epochs = 20, Adam, LR = 1e-3, sparsity = 0. Compare:

- Bits = 8, Int bits = 4 → `ap_fixed<8,4>`
- Bits = 4, Int bits = 2 → `ap_fixed<4,2>`
- Bits = 2, Int bits = 1 → `ap_fixed<2,1>`

Q6 How does reducing the bit-width affect accuracy? At what bit-width does the model become unusable?

Tip hls4ml works best with `ap_fixed<8,4>` or `ap_fixed<16,6>`. The int bits define how many bits represent the integer part (too few and the model saturates).

3.2. QAP: combining quantization and pruning

Fix bits = 8, int bits = 4, epochs = 20, Adam, LR = 1e-3. Run **QAP (QKeras)** varying sparsity:

- Sparsity = 0.0 (quantization only)
- Sparsity = 0.4
- Sparsity = 0.6

Sparsity should be configured in the Pruning tab.

Q7 Does adding pruning on top of quantization hurt accuracy more than each technique alone? Compare the Sparsity metric in the metrics panel for each run.

Part 4: Model Comparison & Score

Before running the pipeline, make sure to load both architectures in the **Architecture** tab:

- **Student** editor → Save → Load into Pipeline
- **QKeras** editor → Save → Load into Pipeline → toggle to **External model** in the QAP tab

Then select **KD** → **QAP (QKeras)** in the Pipeline tab and click **Run Pipeline**.

4.1. Score priorities

Run the pipeline and observe the comparison table. Then change the Score Priorities:

- Preset: Accuracy, which model wins?
- Preset: FPGA, which model wins?
- Preset: Embedded, which model wins?

Q9 Does the same model win under all three presets? What does this tell you about the tradeoffs?

Summary: Key Concepts

Technique	What it does	Main tradeoff
Baseline training	Trains a full-precision float model	Accuracy vs overfitting (epochs, LR)
Pruning	Zeros out low-magnitude weights	Sparsity vs accuracy drop
QAT (QKeras)	Trains with fixed-point quantization	Bit-width vs accuracy
QAP (QKeras)	Quantization + pruning together	Max compression vs accuracy loss
KD + QAP Student	Distills teacher into quantized+pruned student	Knowledge transfer vs compression
Score	Ranks models by weighted criteria	Accuracy vs size vs params