Pebbles vs Shells Image Classification

I. Training Phase

In this phase, the script train_max78000.sh executes the main train.py in which this is configured in order to perform the Quantization-Aware Training. The table below shows the correct folder structure for the dataset for pebbles and shells.

	Dataset	**Data Loader**	**Model**	**Train Script**
Folder	data	datasets	models	scripts
File	pebbles_vs_shells	pebbles_vs_shells.py	ai85net-cd.py	Train_max78000.sh

Table. *Folder Structure*

- The process begins with the load datasets where it first calls the pebbles_vs_shells.py loader to fins the dataset in the directory *data/pebbles_vs_shells*. This directory splits into train and test folders where all of the images are resized to 128x128 pixels and the training images are passed through an albumentations pipeline for the data augmentation.

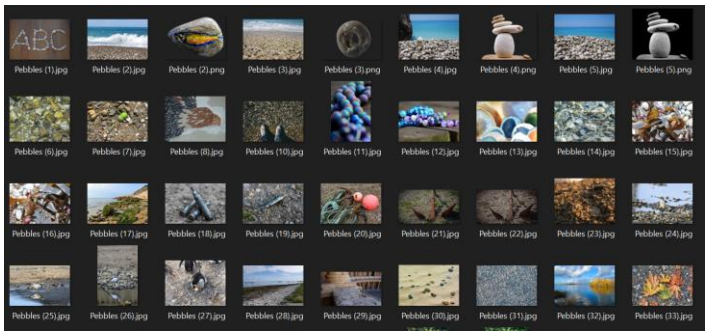
```
# Training Script for Pebbles and Shells
python train.py --epochs 100 --optimizer Adam --lr 0.001 --wd 0 --deterministic \
--compress policies/schedule-pebblesshells.yaml --model ai85cdnet --dataset pebbles_vs_shells \
--confusion --param-hist --embedding --device MAX78000 --workers 0 --enable-tensorboard
```

Code Snippet for train_max78000.sh

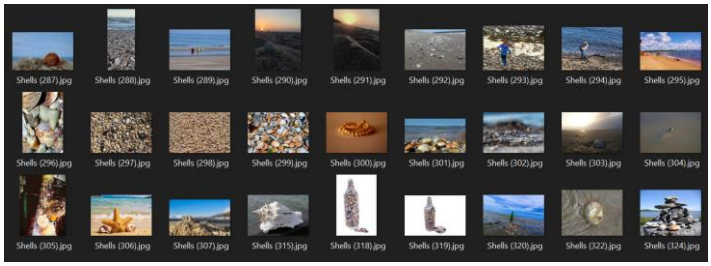
- This trains a machine learning model to classify images of pebbles and shells using the "ai85cdnet" architecture and the "pebbles_vs_shells" dataset. It runs for 100 epochs with the Adam optimizer and includes tools and for evaluating the system performance.

```
Epoch: [99][   12/   12]    Overall Loss 0.461397    Objective Loss 0.461397    Top1 75.456389    LR 0.000216    Time 5.309154
--- validate (epoch=99)-----------
339 samples (256 per mini-batch)
Epoch: [99][    2/    2]    Loss 0.537802    Top1 75.516224
==> Top1: 75.516    Loss: 0.538

==> Confusion:
[[193  33]
 [ 50  63]]

==> Best [Top1: 77.286   Sparsity:0.00   Params: 57776 on epoch: 65]
Saving checkpoint to: logs\2025.11.07-230746\qat_checkpoint.pth.tar
```

Display Output Result of Training Phase



Pebbles Dataset Images

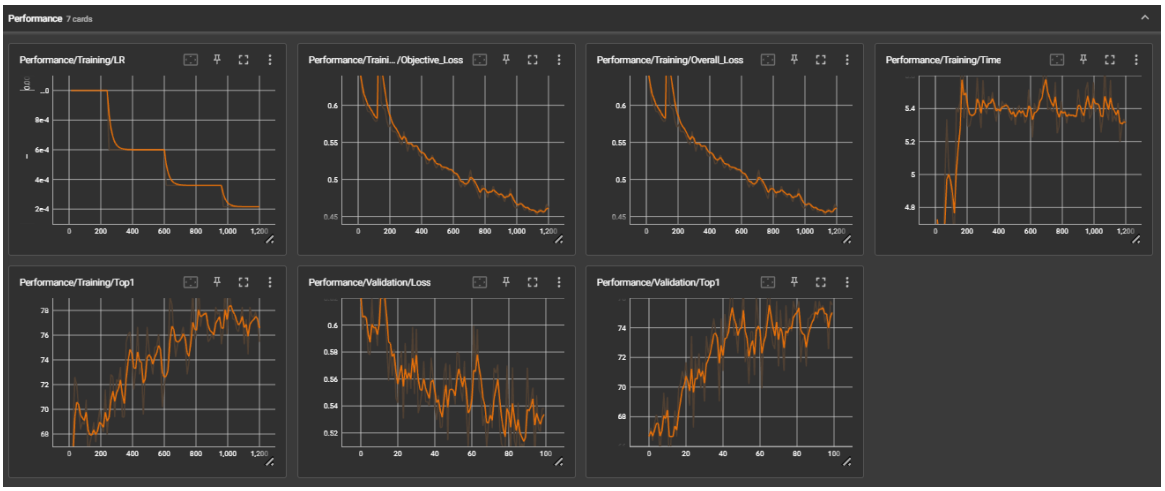


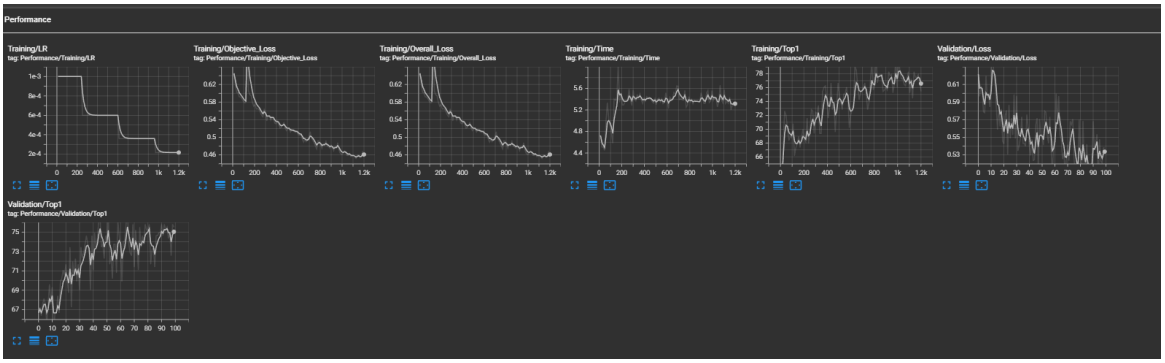
Shells Dataset Images

II. Log Display on TensorBoard

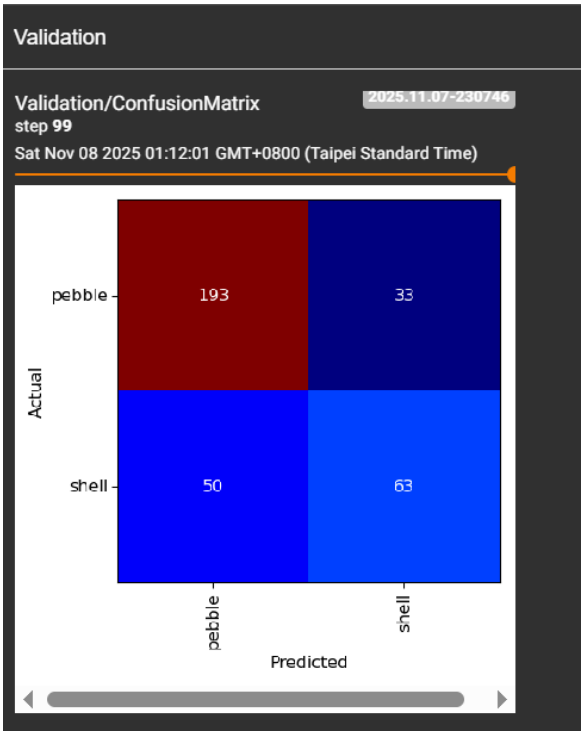
After the training phase, the performance data is saved to the *./logs* directory, where this data can be visualized using the TensorBoard allowing the user view the training progress.

- *Performance Training*





- *Confusion Matrix*



- This is the confusion matrix of the pebbles vs shells showing the correct and incorrect guesses. The top-left shows the high percentage for correctly guessed as Pebbles. The top-right us the incorrect prediction of the model which the model saw a pebble but predicted as shell. For the bottom-left, the model saw a shell but incorrectly predicted it was a pebble. And for the bottom-right, is the correct prediction of the model which is a shell.

III. Quantization

The best performing model checkpoint was copied to the logs folder to the synthesis folder and renamed to ai85-pebblesshells-qat8-q.pth.tar.

```
Ian James B. Cruza@IJBC-13 MINGW64 /c/ai8x/ai8x-synthesis (develop)
$ source scripts/quantize_pebblesvsshells.sh
Configuring device: MAX78000
Converting checkpoint file trained/ai85-pebblesshells-qat8-q.pth.tar to trained/ai85-pebblesshells-qat8-q.pth.tar

Model keys (state_dict):
conv1.output_shift, conv1.weight_bits, conv1.bias_bits, conv1.quantize_activation, conv1.adjust_output_shift, conv1.shift_quantile, conv1.op.weight, conv2.output_shift, conv2.weight_bit
s, conv2.bias_bits, conv2.quantize_activation, conv2.adjust_output_shift, conv2.shift_quantile, conv2.op.weight, conv3.output_shift, conv3.weight_bits, conv3.bias_bits, conv3.quantize_a
ctivation, conv3.adjust_output_shift, conv3.shift_quantile, conv3.op.weight, conv4.output_shift, conv4.weight_bits, conv4.bias_bits, conv4.quantize_activation, conv4.adjust_output_shift
, conv4.shift_quantile, conv4.op.weight, conv5.output_shift, conv5.weight_bits, conv5.bias_bits, conv5.quantize_activation, conv5.adjust_output_shift, conv5.shift_quantile, conv5.op.wei
ght, conv6.output_shift, conv6.weight_bits, conv6.bias_bits, conv6.quantize_activation, conv6.adjust_output_shift, conv6.shift_quantile, conv6.op.weight, fc.output_shift, fc.weight_bits
, fc.bias_bits, fc.quantize_activation, fc.adjust_output_shift, fc.shift_quantile, fc.op.weight, fc.op.bias
conv1.op.weight avg_max: 0.24255072 max: 0.4131112 mean: -0.0066598617 factor: [256.] bits: 8
conv2.op.weight avg_max: 0.23264983 max: 0.33629292 mean: -0.0046767425 factor: [256.] bits: 8
conv3.op.weight avg_max: 0.18263893 max: 0.24982055 mean: -0.0037277412 factor: [512.] bits: 8
conv4.op.weight avg_max: 0.28357205 max: 0.34654617 mean: -0.0034813574 factor: [256.] bits: 8
conv5.op.weight avg_max: 0.25106442 max: 0.35027996 mean: -0.0019362378 factor: [256.] bits: 8
conv6.op.weight avg_max: 0.35907176 max: 0.4151516 mean: -0.0029514986 factor: [256.] bits: 8
fc.op.weight avg_max: 0.09052715 max: 0.09851029 mean: -0.0003753866 factor: [1024.] bits: 8
fc.op.bias avg_max: 0.003403171 max: 0.023830494 mean: -0.003403171 factor: [1024.] bits: 8
```

- This quantize_pebblesvsshells.sh script was executed to synthesize the model. Which this phase it loads the 8-bit quantized checkpoint, converts the model weights and biased into specific format, and the verbose output that confirms the layers that successfully analyzed.

IV. Evaluation

The script evaluate_max78000.sh was run to get the final performance score for the model.

```
$ source scripts/evaluate_max78000.sh
Configuring device: MAX78000, simulate=True.
Log file for this run: C:\ai8x\ai8x-training\logs\2025.11.08-023632\2025.11.08-023632.log
{'start_epoch': 10, 'weight_bits': 8}
=> loading checkpoint ../ai8x-synthesis/trained/ai85-pebblesshells-qat8-q.pth.tar
=> Checkpoint contents:
+----------------------+-------------+-----------+
| Key                  | Type        | Value     |
|----------------------+-------------+-----------|
| arch                 | str         | ai85cdnet |
| compression_sched    | dict        |           |
| epoch                | int         | 99        |
| extras               | dict        |           |
| optimizer_state_dict | dict        |           |
| optimizer_type       | type        | Adam      |
| state_dict           | OrderedDict |           |
+----------------------+-------------+-----------+

=> Checkpoint['extras'] contents:
+-----------------+--------+-------------------+
| Key             | Type   | Value             |
|-----------------+--------+-------------------|
| best_epoch      | int    | 65                |
| best_mAP        | int    | 0                 |
| best_top1       | float  | 77.28613569321534 |
| clipping_method | str    | MAX_BIT_SHIFT     |
| current_mAP     | int    | 0                 |
| current_top1    | float  | 75.51622418879056 |
+-----------------+--------+-------------------+
```

```
Loaded compression schedule from checkpoint (epoch 99)
=> loaded 'state_dict' from checkpoint '../ai8x-synthesis/trained/ai85-pebblesshells-qat8-q.pth.tar'
Optimizer Type: <class 'torch.optim.sgd.SGD'>
Optimizer Args: {'lr': 0.1, 'momentum': 0.9, 'dampening': 0, 'weight_decay': 0.0001, 'nesterov': False}
Dataset sizes:
        training=3053
        validation=339
        test=849
--- test ---------------------
849 samples (256 per mini-batch)
==> Saving sample at index 1 to sample_pebbles_vs_shells.npy
Test: [    4/    4]    Loss 0.527790    Top1 73.969376
==> Top1: 73.969    Loss: 0.528

==> Confusion:
[[492  48]
 [173 136]]
```

- This loads the fully quantized model and runs the model against the entire dataset. The result shows that the Top-1 Accuracy of 73.969376. This is a slight drop from the 77.2861% training peak is an expected and normal trade-off for converting a 32-bit model into 8-bit model.

V. Why it didn't reach 80% accuracy?

- I think the cause of this is the dataset is kind of hard for it to classify as the pebbles and shells are somewhat similar to shape and size which the training accuracy percentage is quite not that high. The model used also is too simple to learn the complex pattern in the dataset, as this

model is fast. I also did try to make the dataset image size to 224x224 with models that can increase the accuracy, however, it did not complete and failed with assertion error. In conclusion, this training with 100 epoch was successful but this only reaches the accuracy of 77.2861%, which I think that is the maximum possible accuracy for this model using the pebbles and shells dataset.

VI. GitHub Link

https://github.com/ianjamescruza/COE187.git