GREENWAVES TECHNOLOGIES

# GAP9
# Tensor Compression

November 20, 2021

Version 1.0

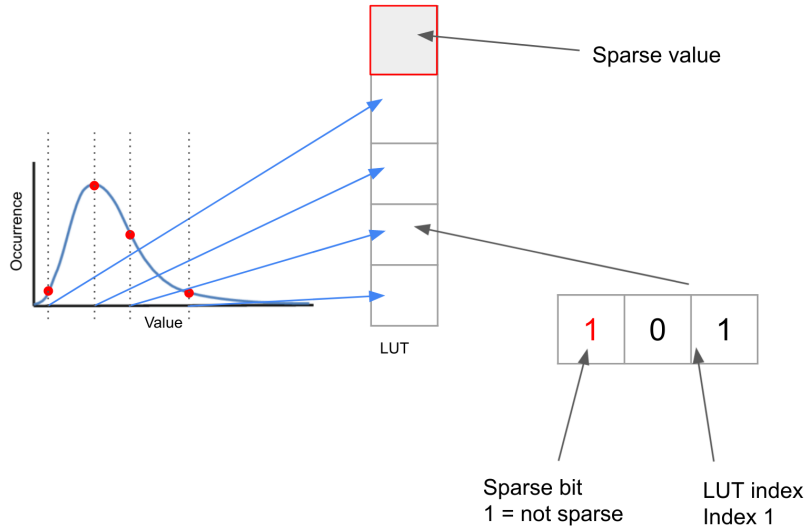
# GAP9 Tensor Compression

GAP9 is equipped with a tensor (de)compression engine integrated into the cluster DMA. The engine can be used to reduce the memory required in L2/L3 for parameter tensors and improve inference speed by reducing L3/L2 and L2/L1 memory transfer size. THis document looks at the hardware operation and the facilities in NNTOOL to experiment with this feature.

The engine can work in 3 modes.

1. Bit packing or unpacking with sign extension
2. Lookup table (LUT) based tensor decompression where the index into the LUT can be from 2 to 8 bits long and the values 8, 16 or 32 bit.
3. Sparse mode. Each LUT index is preceded by a single bit. If 0 this indicates a sparse value and no index follows it. If it is one then it is followed by a 2-8 bit LUT idex as in mode 2.

Modes 2 and 3 are intended for parameter decompression. The fact that the lookup table values are independent of each other means that the points can be chosen to optimally quantize the distribution of the encoded parameter. I.e.  table values can be unevenly distributed on the optimal centroids of the tensor values. This reduces the quantization error improving network accuracy.




The number of entries in the lookup table will be equal to $2^n$ where n is the number of bits selected. If the sparse mode is used then the number of entries will be $2^n + 1$, one extra entry for the sparse element.

Using the compressor the constant tensors in a neural network are stored and transferred from L2 and L3 memory in their compressed format and only expanded at the point that they are written by the cluster DMA in L1 memory. This can significantly reduce the amount of data transferred, both improving performance and reducing energy usage.

The compressed tensor size will be ceiling((n bits * tensor size)/8) + $2^n$ * original quantized byte size. In sparse mode non-sparse elements take up 1 bit more but sparse elements take

up only one bit. Sparse mode also requires some extra storage for tile offsets in the generated autotiler code. This means that sparse mode should not be used unless the tensor is sufficiently sparse (approximately at least 1/n elements are the sparse value).

NNTOOL allows experimentation with this feature using the compress command. The compress command allows the construction of the lookup table and clamps parameter values to the discovered non-uniform centroids of the distribution of the parameter when executing the network. This allows the network accuracy to be evaluated, quantized or not, with the compressed parameter and the error induced to be calculated. If the network is quantized then the table values will be the quantized version of the centroids. If not quantized then they will be single precision float values.

The compressor is compatible with all software kernels[1] available for GAP9 but is not compatible with the NE16 accelerator.

Let's look at the help for the compress command:

```
(NNT model_quantized.tflite 0) help compress
Usage: compress [-h] [--no_sparse] [--force_sparse] [--threshold THRESHOLD] [step] {bits,
min_qsnr, auto, clear, off, on, save, load} ...

Compress graph constants for GAP9 compression engine. Compress with no arguments will list
current compression settings.
The compressed size in the results table includes the size of the codebook.

In bits mode the amount of bits used for table indexes is specified directly. It should be
from 2 to 8.
In min_qsnr mode the tensors will be compressed with the number of bits necessary to stay
above the given QSNR value. A
value of around 30 is a good starting point.

The auto mode uses the validation engine to explore possible tensor compression parameters.
THe command options are the
same as the validate command. The first part of the process tries to find the lowest QSNR that
can be selected for
compression of all viable parameters with no bad validation results.

The threshold argument can be used to clip values to zero before compression.

positional arguments:
  step                  constant input to compress. A list of nodes. You can select nodes with
name, step(negative counts from the end), name wildcard ending with * or a step range e.g. 1:
3

optional arguments:
  -h, --help            show this help message and exit
  --no_sparse           Do not check for sparsity
  --force_sparse        Force these layers to use sparse bit (adds an extra bin)
  --threshold THRESHOLD
                        set values val>x>-val to 0 before clustering

compress subcommands:
  {bits, min_qsnr, auto, clear, off, on, save, load}
                        compression strategy for the selected layers
    bits                compress using a lookup index of a fixed number of bits
    min_qsnr            compress keeping the value QSNR above a minimum value
```

[1] Compressor support is not yet available in the AutoTiler. We expect to implement initial support without sparsity in Q1 2022. At present the accuracy and error induced can be evaluated in NNTOOL only.

```
    auto                compress to a number of bits automatically using validation results
    clear               clear compression on these nodes
    off                 disable compression on these nodes
    on                  enable compression on these nodes
    save                save compression settings to a file in json format
    load                load compression settings from a file in json format
```

```
(NNT) help compress bits
Usage: compress bits [-h] {2, 3, 4, 5, 6, 7}

positional arguments:
  {2, 3, 4, 5, 6, 7}  number of bits to use for lookup table indexes

optional arguments:
  -h, --help          show this help message and exit

(NNT) help compress min_qsnr
Usage: compress min_qsnr [-h] qsnr

positional arguments:
  qsnr        QSNR to keep above

optional arguments:
  -h, --help  show this help message and exit
```

The compress command has 3 sub commands: bits, min_qsnr and auto. In *bits* mode the exact amount of bits used for the table indexes is specified explicitly. In *min_qsnr* mode the clustering is repeated on an increasing amount of bits from 2-8 until the QSNR between the original tensor and the compressed tensor exceeds the given value. In *auto* mode NNTOOL tries to find optimal compression parameters using the same validation process as the *validation* command. This is useful for finding a global compression point that can be further tuned manually.

In all the modes a threshold can be provided to clamp tensor elements below a certain absolute value to 0 improving the chance of finding a sparse value of 0. Sparsity must be at a certain level for it to actually compress the tensor further.

The compress command uses an efficient 1D implementation[2] of the k-means clustering algorithm to discover the centroids of the $2^n$ (or $2^n + 1$) clusters.

The process we have found the most efficient for discovering optimal compression levels is first to tune all parameters to the lowest QSNR that causes no accuracy loss and then to fine tune large parameters that remain validating the graph using the *validate* command on a small sample of data in between each experiment. We anticipate automating this search algorithm as we gain more empirical evidence that it works for different network types. Compression automation could also be implemented as part of Quantization Aware Training (QAT) process. Generally the validation can first be done on float (i.e. not quantized) values however there can be small differences with compressed quantized evaluation so each tuning step should be validated with quantized execution and the end.

An example

[2] See https://github.com/dstein64/kmeans1d & https://arxiv.org/pdf/1701.07204.pdf

Let's look at how this works in practice using the visual wakewords example. First open the graph with nntool and adjust and fuse.

```
$ nntool examples/nntool/visual_wake/model/visual_wake_quant.tflite -q

(NNT visual_wake_quant.tflite 0) adjust

(NNT visual_wake_quant.tflite 0) fusions --scale8
```

Set an input normalization function:

```
(NNT visual_wake_quant.tflite 0) set input_norm_func 'x:x/128-1'
```

Now we can start the compression process. First we use the auto mode to find a global QSNR setting.

```
(NNT visual_wake_quant.tflite 0) compress * auto examples/nntool/visual_wake/images/* -T
validation without compression : ++++++++++ <- First the validation is run without compression
good 10 (0.90) bad 0 (0.00)
QSNR 30 step 15 direction down <- Now a test is run at 30 QSNR
compressing: --+-+-+-+++++-+-+++-++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++-
validating: ++++++++++
good 10 (0.93) bad 0 (0.00) <- Validation passes so it continues to reduce the QSNR
QSNR 15 step 15 direction down
compressing: ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++-
validating: ++--+--+-+
good 5 (0.78) bad 5 (0.71) <- At a QSNR or 15 there are failures so it starts to move up.
QSNR 22.5 step 7.5 direction up
compressing: +-+++++-++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++-
validating: +++++  <- Only the inputs that caused errors are validated
good 5 (0.49) bad 0 (0.00)
QSNR 18.75 step 3.75 direction down
compressing: +-++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++-
validating: +----
good 1 (0.03) bad 4 (0.59)
QSNR 20.625 step 1.875 direction up
compressing: +-++++++-++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++-
validating: ----
good 0 (0.00) bad 4 (0.82)
QSNR 22.5 step 1.875 direction up
QSNR 21.5625 step 0.9375 direction down
compressing: +-+++++-++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++-
validating: +++-
good 3 (0.32) bad 1 (0.07)
QSNR 22.03125 step 0.46875 direction up
compressing: +-+++++-++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++-
validating: + <- Everything is validated at 22.5
good 1 (0.13) bad 0 (0.00)
check with all inputs
QSNR 22.5 step 0.5 direction up
compressing: +-+++++-++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++-
validating: ++++++++++
good 10 (0.62) bad 0 (0.00)
tune QSNR to best 22.5 compressed by 97355 bytes
+-+++++-++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++-
Compression report <- The compression report shows that the parameters are at 64% of their original size with no validation errors
```

| Step | Name | Orig Size | Compressed Size | %age orig | Bits | Sparse | Enabled |
|---|---|---|---|---|---|---|---|
| 1 | first_convconvweights_quantFak | 216 | 167 | 77 | 5 | No | Yes |
| 4 | blocks0mobile_inverted_convinv_607a0a32 | 96 | 64 | 66 | 4 | No | Yes |
| 5 | blocks0mobile_inverted_convinv_fd1dc8c9 | 48 | 36 | 75 | 3 | No | Yes |
| 7 | blocks0mobile_inverted_convdep_e3322ef4 | 108 | 70 | 64 | 4 | No | Yes |
| 8 | blocks0mobile_inverted_convdep_8ca263cc | 48 | 36 | 75 | 3 | No | Yes |
| 10 | blocks0mobile_inverted_convpoi_fe2eb3db | 96 | 64 | 66 | 4 | No | Yes |
| 13 | blocks1mobile_inverted_convinv_122d1efa | 160 | 132 | 82 | 5 | No | Yes |
| 14 | blocks1mobile_inverted_convinv_381f246e | 80 | 39 | 48 | 3 | No | Yes |
| 16 | blocks1mobile_inverted_convdep_ed934395 | 500 | 344 | 68 | 5 | No | Yes |

```
| 17   | blocks1mobile_inverted_convdep_e79aca04 | 80        | 39              | 48        | 3    | No     | Yes     |
+------+-----------------------------------------+-----------+-----------------+-----------+------+--------+---------+
| 19   | blocks1mobile_inverted_convpoi_eaf1a309 | 320       | 232             | 72        | 5    | No     | Yes     |
+------+-----------------------------------------+-----------+-----------------+-----------+------+--------+---------+
| 20   | blocks1mobile_inverted_convpoi_5ab9b594 | 64        | 38              | 59        | 3    | No     | Yes     |
+------+-----------------------------------------+-----------+-----------------+-----------+------+--------+---------+
| 22   | blocks2mobile_inverted_convinv_c11e5c4c | 576       | 392             | 68        | 5    | No     | Yes     |
+------+-----------------------------------------+-----------+-----------------+-----------+------+--------+---------+
```

Output omitted ...

```
+------+-----------------------------------------+-----------+-----------------+-----------+------+--------+---------+
| 161  | blocks16mobile_inverted_convde_02290bd9 | 3800      | 2407            | 63        | 5    | No     | Yes     |
+------+-----------------------------------------+-----------+-----------------+-----------+------+--------+---------+
| 162  | blocks16mobile_inverted_convde_e1c96138 | 608       | 140             | 23        | 4    | No     | Yes     |
+------+-----------------------------------------+-----------+-----------------+-----------+------+--------+---------+
| 164  | blocks16mobile_inverted_convpo_1741f005 | 14592     | 9152            | 62        | 5    | No     | Yes     |
+------+-----------------------------------------+-----------+-----------------+-----------+------+--------+---------+
| 165  | blocks16mobile_inverted_convpo_d008e517 | 384       | 112             | 29        | 4    | No     | Yes     |
+------+-----------------------------------------+-----------+-----------------+-----------+------+--------+---------+
| 168  | feature_mix_layerconvweights_q          | 61440     | 38432           | 62        | 5    | No     | Yes     |
+------+-----------------------------------------+-----------+-----------------+-----------+------+--------+---------+
| 169  | feature_mix_layerconvConv2D_bi          | 2560      | 528             | 20        | 5    | No     | Yes     |
+------+-----------------------------------------+-----------+-----------------+-----------+------+--------+---------+
| 171  | classifierlinearweights_quantF          | 1280      | 832             | 65        | 5    | No     | Yes     |
+------+-----------------------------------------+-----------+-----------------+-----------+------+--------+---------+
|      | Total                                   | 238472    | 153069          | 64        |      |        |         |
+------+-----------------------------------------+-----------+-----------------+-----------+------+--------+---------+
```

We can further tune individual layers with the compress command. In classifier networks the final layers are often little more than decision trees. We can try to reduce them down to this by drastically reducing the bits and clamping low values to 0.

```
(NNT visual_wake_quant.tflite 0) compress --threshold 0.05 --force_sparse 168 bits 2

Evaluating feature_mix_layerconvweights_q
Compression report

+------+---------------------------------+-----------+-----------------+-----------+------+--------+---------+
| Step |              Name               | Orig Size | Compressed Size | %age orig | Bits | Sparse | Enabled |
+======+=================================+===========+=================+===========+======+========+=========+
| 168  | feature_mix_layerconvweights_q  | 61440     | 13901           | 22        | 2    | Yes    | Yes     |
+------+---------------------------------+-----------+-----------------+-----------+------+--------+---------+
|      | Total                           | 61440     | 13901           | 22        |      |        |         |
+------+---------------------------------+-----------+-----------------+-----------+------+--------+---------+
```

This time we need to run the validate command manually.

```
(NNT visual_wake_quant.tflite 0) validate auto examples/nntool/visual_wake/images/* -T
validation - quantization mode - none
validation - input file ['examples/nntool/visual_wake/images/COCO_val2014_000000322057_0.ppm']
graph_executer - execute uncached: quantization mode none
validation - Prediction is True predicted 0 correct 0 margin 0.365692
validation - input file ['examples/nntool/visual_wake/images/COCO_val2014_000000362331_0.ppm']
graph_executer - execute uncached: quantization mode none
validation - Prediction is True predicted 0 correct 0 margin 0.7300658
validation - input file ['examples/nntool/visual_wake/images/COCO_val2014_000000174838_1.ppm']
graph_executer - execute uncached: quantization mode none
validation - Prediction is True predicted 1 correct 1 margin 0.63374746
validation - input file ['examples/nntool/visual_wake/images/COCO_val2014_000000145620_1.ppm']
graph_executer - execute uncached: quantization mode none
validation - Prediction is True predicted 1 correct 1 margin 0.510812
validation - input file ['examples/nntool/visual_wake/images/COCO_val2014_000000174091_0.ppm']
graph_executer - execute uncached: quantization mode none
validation - Prediction is True predicted 0 correct 0 margin 0.8696948
validation - input file ['examples/nntool/visual_wake/images/COCO_val2014_000000455859_1.ppm']
graph_executer - execute uncached: quantization mode none
validation - Prediction is True predicted 1 correct 1 margin 0.50327706
validation - input file ['examples/nntool/visual_wake/images/COCO_val2014_000000517417_1.ppm']
graph_executer - execute uncached: quantization mode none
validation - Prediction is True predicted 1 correct 1 margin 0.40557283
validation - input file ['examples/nntool/visual_wake/images/COCO_val2014_000000533261_0.ppm']
graph_executer - execute uncached: quantization mode none
validation - Prediction is True predicted 0 correct 0 margin 0.52109635
validation - input file ['examples/nntool/visual_wake/images/COCO_val2014_000000514089_1.ppm']
graph_executer - execute uncached: quantization mode none
validation - Prediction is True predicted 1 correct 1 margin 0.13810992
validation - input file ['examples/nntool/visual_wake/images/COCO_val2014_000000036861_0.ppm']
graph_executer - execute uncached: quantization mode none
validation - Prediction is True predicted 0 correct 0 margin 0.7127249
```

```
validation - 10 out of 10 predicted correctly with 0.5390793174505234 average margin
validation - Total accuracy: 100.000 %
(NNT visual_wake_quant.tflite 0)
```

All of the samples are predicted correctly. We can use the compress command on its own to print the full report again

```
(NNT visual_wake_quant.tflite 0) compress
Compression report

+------+-----------------------------------------+-----------+-----------------+-----------+------+--------+---------+
| Step |                  Name                   | Orig Size | Compressed Size | %age orig | Bits | Sparse | Enabled |
+======+=========================================+===========+=================+===========+======+========+=========+
| 1    | first_convconvweights_quantFak          | 216       | 167             | 77        | 5    | No     | Yes     |
+------+-----------------------------------------+-----------+-----------------+-----------+------+--------+---------+
| 4    | blocks0mobile_inverted_convinv_607a0a32 | 96        | 64              | 66        | 4    | No     | Yes     |
+------+-----------------------------------------+-----------+-----------------+-----------+------+--------+---------+
| 5    | blocks0mobile_inverted_convinv_fd1dc8c9 | 48        | 36              | 75        | 3    | No     | Yes     |
+------+-----------------------------------------+-----------+-----------------+-----------+------+--------+---------+
| 7    | blocks0mobile_inverted_convdep_e3322ef4 | 108       | 70              | 64        | 4    | No     | Yes     |
+------+-----------------------------------------+-----------+-----------------+-----------+------+--------+---------+
| 8    | blocks0mobile_inverted_convdep_8ca263cc | 48        | 36              | 75        | 3    | No     | Yes     |
+------+-----------------------------------------+-----------+-----------------+-----------+------+--------+---------+
| 10   | blocks0mobile_inverted_convpoi_fe2eb3db | 96        | 64              | 66        | 4    | No     | Yes     |
+------+-----------------------------------------+-----------+-----------------+-----------+------+--------+---------+


+------+-----------------------------------------+-----------+-----------------+-----------+------+--------+---------+
| 164  | blocks16mobile_inverted_convpo_1741f005 | 14592     | 9152            | 62        | 5    | No     | Yes     |
+------+-----------------------------------------+-----------+-----------------+-----------+------+--------+---------+
| 165  | blocks16mobile_inverted_convpo_d008e517 | 384       | 112             | 29        | 4    | No     | Yes     |
+------+-----------------------------------------+-----------+-----------------+-----------+------+--------+---------+
| 168  | feature_mix_layerconvweights_q          | 61440     | 13901           | 22        | 2    | Yes    | Yes     |
+------+-----------------------------------------+-----------+-----------------+-----------+------+--------+---------+
| 169  | feature_mix_layerconvConv2D_bi          | 2560      | 528             | 20        | 5    | No     | Yes     |
+------+-----------------------------------------+-----------+-----------------+-----------+------+--------+---------+
| 171  | classifierlinearweights_quantF          | 1280      | 832             | 65        | 5    | No     | Yes     |
+------+-----------------------------------------+-----------+-----------------+-----------+------+--------+---------+
|      | Total                                   | 238472    | 128538          | 53        |      |        |         |
+------+-----------------------------------------+-----------+-----------------+-----------+------+--------+---------+
```

The graph parameters are now at 53% of their original size