209AS Office hour notes

Shurui Li

Dataflow explanation

- WS: weight stationary, keep weights in the weight buffer as long as possible -> each weight is only loaded once from weight SRAM

- IS: input stationary, keep inputs in the input buffer as long as possible -> each input vector is only loaded once from SRAM

- OS: output stationary, keep outputs/psums in the output buffer as long as possible -> psums are fully accumulated before writing to activation SRAM, so each output is only written once into SRAM

Part 2 WS example

- There are many ways to build your model/simulators
- Here I will provide an example with a simple analytical approach
- WS dataflow, batch size = 1
- Use last layer as example, [512, 256, 3, 3]

Part 2 WS example

- WS dataflow means reuse weights as much as possible -> **each weight only need to be loaded once from SRAM**
- Number of times weight buffer need to be updated (how weights are mapped):
  *ceil(n_filter/16) * ceil(n_channel* filter_size/128)*
  Call this **NWBU**
- Then calculate total dot product unit cycles **NDPC**: Every time the weight buffer is updated, we want to reuse the weights for all **valid** input pixels, therefore:
  *NDPC = NWBU * output_size (8*8 for this layer)*

Now we can calculate the energy results

Weight SRAM energy

- Weight SRAM energy **WSE** is NWBU * total bits loaded into weight buffer * access energy
  *WSE = NWBU * 16 * 128 * weight_bitwidth * SRAM_access_energy*
- You **may** need to adjust for cases when the dot product size is less than 128, not needed for this layer though

Input activation SRAM energy

- Input buffer need to be updated (read from SRAM) for every dot product unit computation (NDPC) , so input activation SRAM energy (**ISE**) can be cauculated using NDPC
  *ISE = NDPC * 128 * activation_bitwidth * SRAM_access_energy*
- Similarly, you **may** need to adjust for cases when the dot product size is less than 128

Output activation SRAM energy

- Output activation SRAM energy is slightly more complicated, as you may need to also load partial sums from SRAM in order to accumulate. For this particular layer, since the actual dot product size (256*9) is larger than the dot product unit size (128), psum will be produced and every time they need to be load from SRAM back to output buffer to accumulate.
- Anyway, output buffer need to write to activation SRAM every time the dot product units produces results, so the output activation SRAM write energy (**OSWE**) is straightforward to calculate:
  *OSWE = NDPC * 16 * activation_bitwidth * SRAM_access_energy*
- If we assume each cycle psum will be read from SRAM to output buffer for accumulation, then output activation SRAM read energy (**OSRE**) is same as OSWE
  OSRE = OSWE

However, for the first psum chunk, no accumulation is needed, how can we adjust for this?

- The original dot product with size 2304 is sliced into 18 chunks of smaller dot products with size 128, and only the first chunk does not need accumulation
- So, $OSRE = \frac{17}{18} OSWE$
- OSE = OSRE + OSWE

Now all SRAM energy is computed, let's compute the total dot product unit energy **DPE**

- This should be straightforward, just use NDPC, which is number of times the dot product units compute
- *DPE = NDPC * 16 * single_dpu_energy_per_cycle*

Putting everything together:

- *Total layer energy = WSE + ISE + OSE + DPE*
- Repeat the above for all layers
- Not done yet… Need to also add DRAM read energy, which is just the total energy required to load all weights from DRAM to SRAM
- Now we get the total energy for WS dataflow with batch size of 1 ☺

Feel free to implement your simulator however you like, either using analytical model or more refined approach