HW 06: k-NN Accelerator Report

104062121 陳品媛

Goal

Implement k-nearest neighbors algorithm (k-NN) with **C and Verilog** on PicoRV32 system.

Implementation

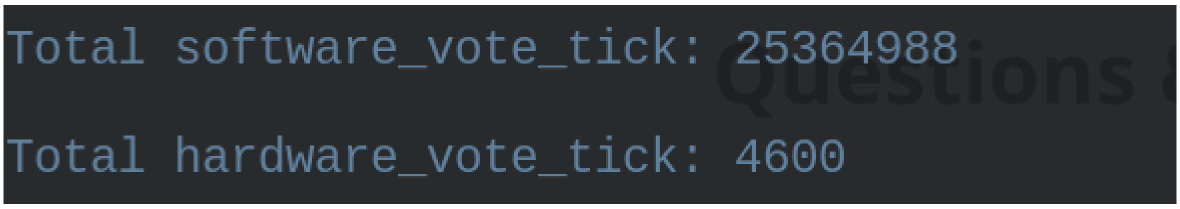
To implement KNN, we have several steps as following listing.

- Catch image pixels and labels
- L2 distance computation
- Sort to get top K shortest distances
- Voted by top K image

Partition

I use **hardware** to **compute distances** between test images and train images because of software's slow speed. And compare the ticks of running **voting** between **software and hardware**.

In terms of voting, software is much slower than hardware.



```
Total software_vote_tick: 25364988

Total hardware_vote_tick: 4600
```

Memory

Both software part computing and hardware part computing need get the **image pixels and image labels**.

Their arrangement is like (label + red + green + blue) for a image total 3073 pixels.

- image pixel
  - IMAGE_OFFSET + (image_index * DATA_LENGTH +pixel_pos)* 4
  - 1 <= pixel_pos <= 3072
- image label
  - IMAGE_OFFSET + image_index * DATA_LENGTH * 4

Distance

The square of L2 (Euclidean) distance.

$\sum(testPixel - trainPixel)^2$

TopK

Pick out the K shortest distance for voting step.

Iterate 950 train image distances from the test image. If we iterate the distance which is smaller than the what we have iterated, then inserted it into the array.

Vote

Only the top k have the right to vote. The final answer is determined by the label which get the most votes. When ending in a tie, I choose the smaller label.

Software

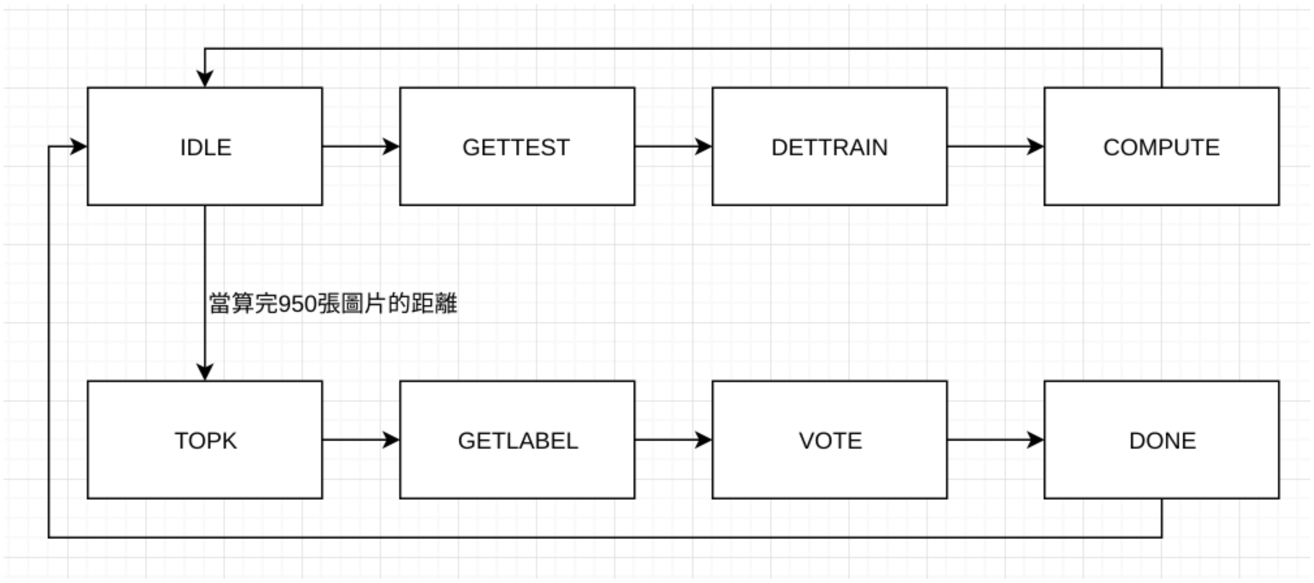
I implement distance computing. And the voting part used is written by TA. There is only a small point I modified is when some labels has the same vote, I choose the smaller label.

Hardware

FSM

1. IDLE: wait for the pcpi_insn_valid =1, then judge going to get pixels or sorting the distances.
2. GETTEST: pass test image pixel's address to get test image pixels in the next cycle
3. GETTRAIN: pass train image pixel's address to get train image pixels in the next cycle
4. COMPUTE: use test image pixel and train image pixel to compute the square of the L2 distance
5. TOPK: sort the distances to get the top k shortest distances and save their image number
6. GETLABEL: pass the top k image's label's address to get the label in the next cycle
7. VOTE: judge which label get the most vote
8. DONE: done the knn

Flow chart



PCPI

- when pcpi_insn_valid = 1'b1, start computing
- when pass the return value

  ```
  pcpi_wr = 1'b1;
  pcpi_wait = 1'b0;
  pcpi_ready = 1'b1;
  pcpi_rd = max_label;
  ```

- when unfinish the computation

  ```
  pcpi_wr = 1'b0;
  pcpi_wait = 1'b1;
  pcpi_ready = 1'b0;
  pcpi_rd = 32'd0;
  ```

Memory interface for PicoRV32

All the operations done in this homework are **reading the data** from the memory.

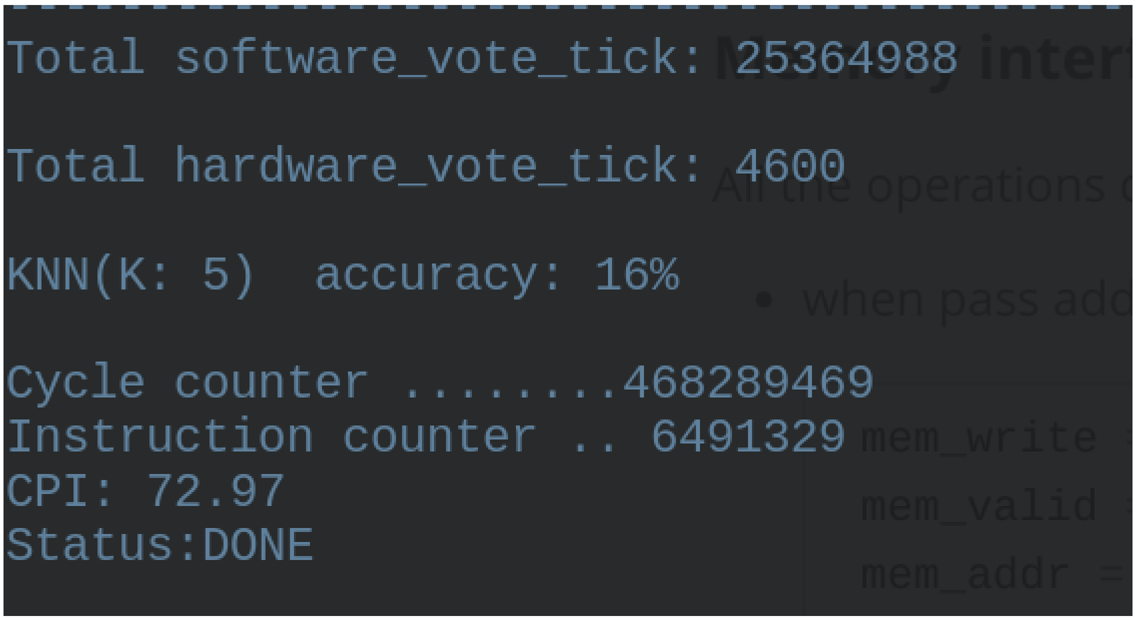
- when pass address

  ```
  mem_write = 1'b0
  mem_valid = 1'b1
  mem_addr = data_addr
  ```

Questions & Discussion

1. How do you partition your algorithm?
   - hardware to compute the distance
   - i could choose use software or hardware to finish the rest part(find topk, voting)
2. Profile both software and hardware versions. Which part do you think is the bottleneck of your (software/hardware) design?

   computing L2 distance, which really cost a lot of time
3. Briefly describe the experiments you did and their results.
   - modify K to 5



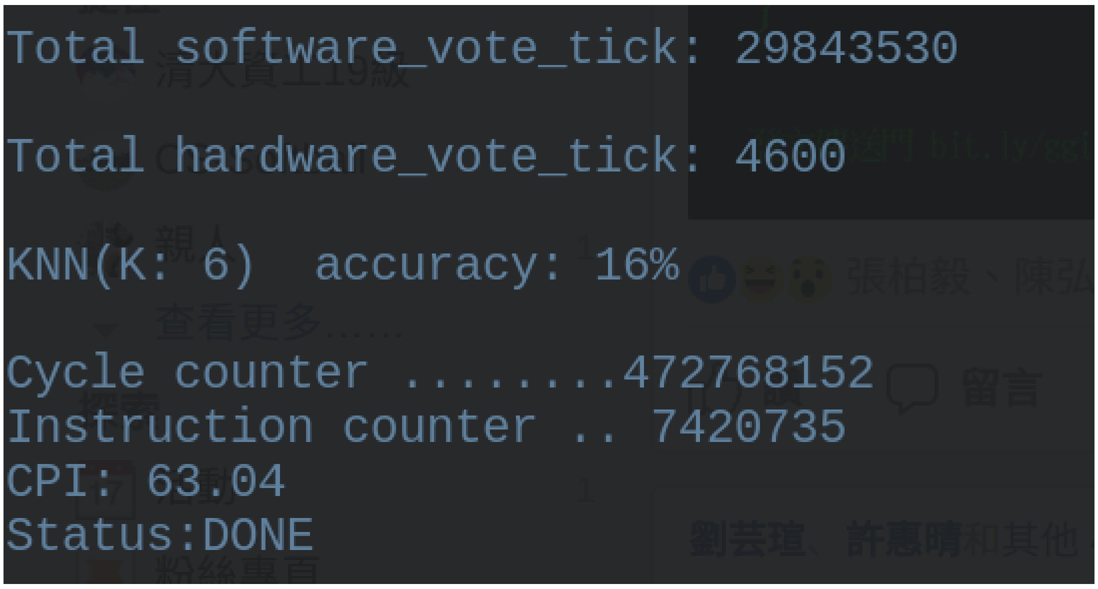
```
Total software_vote_tick: 25364988

Total hardware_vote_tick: 4600

KNN(K: 5)  accuracy: 16%

Cycle counter .........468289469
Instruction counter .. 6491329
CPI: 72.97
Status:DONE
```

- modify K to 6



```
Total software_vote_tick: 29843530

Total hardware_vote_tick: 4600

KNN(K: 6)  accuracy: 16%

Cycle counter .........472768152
Instruction counter .. 7420735
CPI: 63.04
Status:DONE
```